

Parte 1: Introduzione ai comandi UNIX

(<https://www.youtube.com/watch?v=ypEaGQb6dJk>)



Lezione 3

Interfacce utente

Sistemi Operativi (9 CFU), CdL Informatica, A. A. 2022/2023

Dipartimento di Scienze Fisiche, Informatiche e Matematiche

Università di Modena e Reggio Emilia

<http://weblab.ing.unimo.it/people/andreolini/didattica/sistemi-operativi>

Quote of the day

(<http://www.cryptonomicon.com/beginning.html>)

“Windows 95 and MacOS are products, contrived by engineers in the service of specific companies. Unix, by contrast, is not so much a product as it is a painstakingly compiled oral history of the hacker subculture.”

Neal Stephenson (1959-)

Scrittore

Autore di “In the beginning was the command line”



Lo scenario

(Neanche tanto sbagliato, a pensarci bene)

Uno studente decide di scaricare, installare e provare ad usare il SO GNU/Linux.

Scuse legittime per usare GNU/Linux:

- si è costretti dal docente.
- si è costretti dal datore di lavoro.
- ci si annoia e non sa che fare.
- si nutre una sincera curiosità.
- le cavallette.



Interrogativi 1/2

(Individuare e conoscere le interfacce d'uso a disposizione)

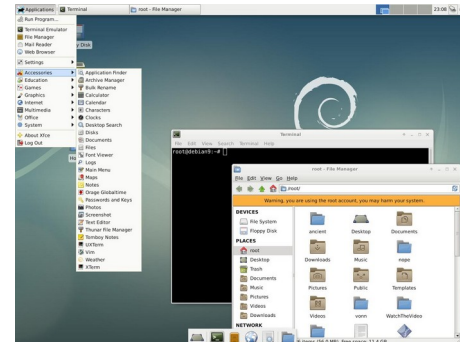
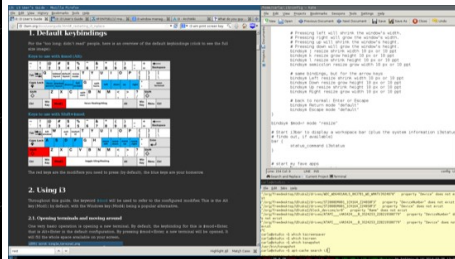
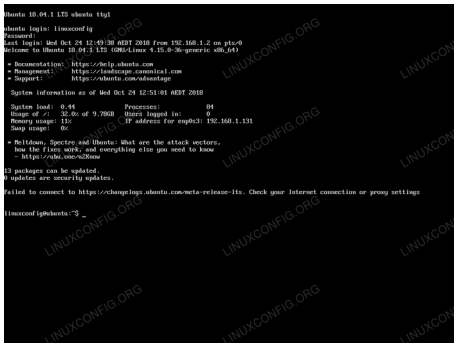
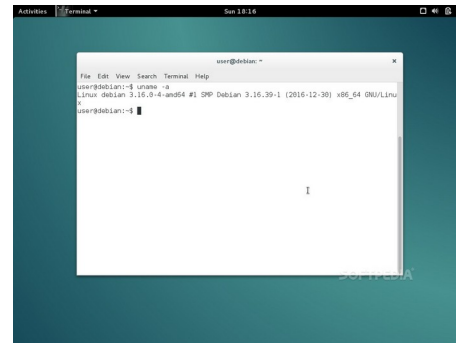
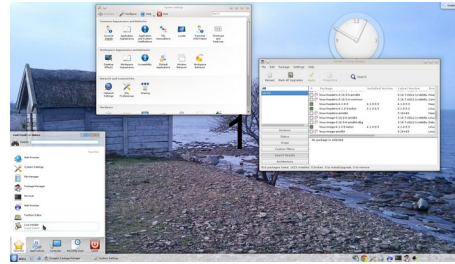
Quali interfacce utente mette a disposizione GNU/Linux?

Interfacce di tipo desktop?

Interfacce testuali?

Altro?

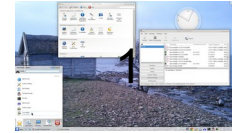
Che cosa si intende esattamente per "interfaccia utente"?



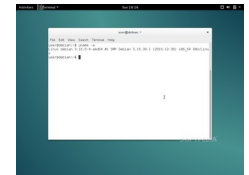
Interrogativi 2/2

(Valutare i pro e i contro di ciascun tipo di interfaccia d'uso)

Quali sono i pro e i contro delle diverse categorie di interfacce utente?



Su quale tipologia di interfaccia utente dovrebbe concentrarsi l'utente desideroso di approfondire la conoscenza di GNU/Linux?



INTERFACCE UTENTE

Interfaccia utente

(Permette di comunicare con il calcolatore)

L'**interfaccia utente** di un SO è l'insieme di meccanismi con i quali avviene l'interazione uomo-macchina.

Interazione:

l'utente specifica una operazione.

il SO traduce l'operazione in applicazioni da eseguire.

l'output delle operazioni è reso disponibile all'utente.

Esistono diverse categorie di interfacce utente:

testuali (**Command Line Interface, CLI**).

grafiche (**Graphical User Interface, GUI**).

touchscreen (**Touch User Interface, TUI**).

Caratteristiche salienti delle interfacce

(Comandi, icone, cartelle, menu, gesture)

CLI: uso di un **interprete dei comandi (shell)** per la lettura e l'esecuzione di singoli comandi (**shell interattiva**) o di un gruppo di comandi memorizzati in un file (**shell non interattiva** che esegue uno **script**). I comandi disponibili sono forniti dalla shell oppure dalle utility di base del SO.

GUI: esposizione di una **scrivania virtuale (desktop)**, pilotata da opportuni dispositivi di puntamento (mouse). Le operazioni disponibili sono accessibili tramite menu.

TUI: versione semplificata della GUI. Il mouse è sostituito da un sistema di riconoscimento basato su gesti della mano (**gesture**).

Pro e contro delle GUI/TUI

(Unicuique suum)

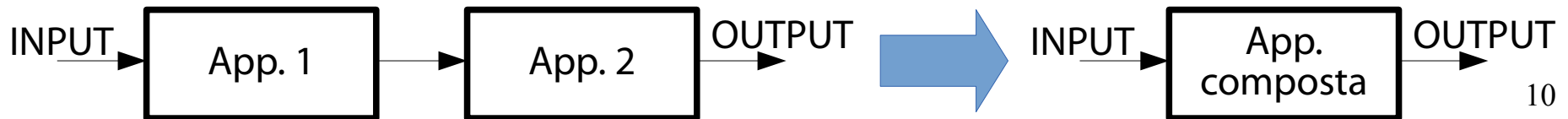
Pro:

molto semplici ed intuitive nell'uso.

Contro:

non sono facilmente componibili.

Componibilità: l'atto di poter “agganciare” due applicazioni in modo tale da creare una nuova applicazione composta.



Condizioni necessarie

(Per la componibilità di GUI/TUI)

Le applicazioni devono usare un formato intercambiabile di rappresentazione dei dati intermedi.

L'output in uscita da App. 1 deve poter essere leggibile da App. 2.

App. 1 ed App. 2 devono potersi “connettere” fra loro.

Tramite strumenti di comunicazione standard quali file, socket di rete, memoria condivisa, altro.

Un esempio di non componibilità 1/2

(Stampa dei file contenuti in una directory)

Provate ad individuare una applicazione basata su GUI/TUI (fra quelle disponibili) che stampi l'elenco di una directory "grande" (contenente, ad esempio, 10000 file).

Esiste una tale applicazione?

In generale, NO.

Se esiste, segnalatela al docente :-)

Un esempio di non componibilità 2/2

(Stampa dei file contenuti in una directory)

Si può provare a comporre le applicazioni:

“file manager grafico”.

“client grafico di stampa”.

La domanda è: COME?

Il file manager grafico non ha modo di costruire un file di testo da inviare al client di stampa.

Quand'anche riuscisse, il file manager non è in grado di “connettersi” al client di stampa e fornirgli il file.

Tali applicazioni sono componibili?

→ In questo caso, NO.

Come si risolve il problema con le GUI?

(Sempre la solita stampa dei file contenuti in una directory)

Soluzioni al problema.

Scrivere una applicazione GUI/TUI “ad-hoc” che svolga il compito specifico (l'applicazione non è riciclabile in seguito).

Produrre manualmente screenshot dei file e stamparli, sempre manualmente (soluzione ripugnante perché non automatica; l'utente lavora per il calcolatore, quando invece dovrebbe essere il contrario!).

Pro e contro delle CLI

(Unicuique suum)

Pro:

sono ideate per essere componibili.

Contro:

Presentano una curva di apprendimento superiore rispetto alle GUI/TUI.

Come si risolve il problema con le CLI?

(Sempre la solita stampa dei file contenuti in una directory)

Basta digitare due comandi.

```
cd /directory/di/interesse
```

```
ls | lpr
```

Entra nella
directory di
interesse

Elenca
i file e le
directory

Passa l'output
di `ls` a `lpr`

Invia l'elenco
al client di
stampa

I due comandi possono essere inseriti in uno script ed eseguiti in modalità batch, senza intervento umano.
→ È il calcolatore a lavorare per voi!

GUI/TUI IN GNU/LINUX

Modularità delle GUI

(Architettura modulare; componenti interscambiabili fra loro)

L'architettura di una GUI in GNU/Linux è **modulare**.

Insieme di componenti software (applicazioni, librerie) cooperanti tra loro.

Ogni applicazione implementa uno specifico aspetto.

Per uno specifico componente possono esistere diverse implementazioni e configurazioni, spesso interscambiabili fra loro.

I componenti di una GUI

(**Server**, window manager, toolkit, applicazione, graphical display manager)

Server grafico.

Implementa un protocollo di comunicazione client-server con gli elementi grafici (in primis, le finestre).

Riceve richieste di disegno.

Trasforma le richieste di disegno in comandi da impartire alla scheda video.

Gestisce i buffer di uno o più schermi.

Gestisce le periferiche di input (mouse, tastiera).

I componenti di una GUI

(**Server**, window manager, toolkit, applicazione, graphical display manager)

XOrg.

Progetto iniziato nel 2004.

Successore di XFree86 (implementazione free di X11, il server grafico UNIX originario).

Implementa le funzionalità descritte nella slide precedente.



Il logo di XOrg

Curiosità

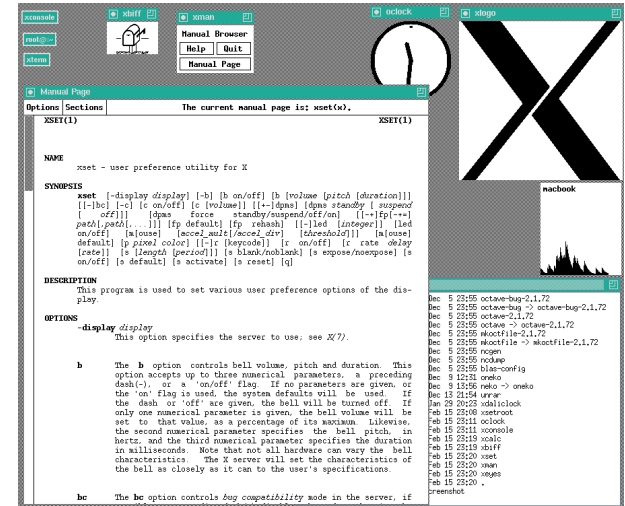
(<https://www.youtube.com/watch?v=SWQy2r6kEF0>)

Perché il prefisso “X”?

XFree86 deriva da X Window System, versione 11 (X11).

X Window System deriva da W Window System, sviluppato a Stanford nel 1981 per il sistema operativo V.

(X è la lettera successiva a W).



X Window System

I componenti di una GUI

(**Server**, window manager, toolkit, applicazione, graphical display manager)

Pro.

Software maturo, consolidato e testato.

Contro.

Prestazioni non ottimali.

Il server converte tutte le richieste di disegno delle finestre in pixel (**rendering**), ed invia i buffer alla scheda video.

→ Collo di bottiglia (più richieste di vengono inviate, più il sistema si rallenta).



Il logo di XOrg

I componenti di una GUI

(**Server**, window manager, toolkit, applicazione, graphical display manager)

Pro.

Software maturo, consolidato e testato.

Contro.

Scarsa attenzione alla sicurezza. Gli elementi grafici non sono isolati fra loro (violato uno, violati tutti). Il server deve eseguire con privilegi elevati per interagire con la scheda video. L'autenticazione è debole (l'utente ottiene un "magic cookie" che, se rubato, permette l'accesso a tutte le finestre).



Il logo di XOrg

I componenti di una GUI

(**Server**, window manager, toolkit, applicazione, graphical display manager)

Wayland.

Progetto iniziato nel 2008.

Protocollo di comunicazione client-server (implementazione di riferimento: **Weston**).

Cerca di risolvere i problemi di Xorg.



Il logo di Wayland

Curiosità

(<https://www.youtube.com/watch?v=rDMBo19wllw>)

Perché “Wayland”?

Kristian Høgsberg, creatore del progetto, viaggiava in macchina per le vie di Wayland (Massachusetts) quando ebbe l’illuminazione.



Wayland, MA

I componenti di una GUI

(**Server**, window manager, toolkit, applicazione, graphical display manager)

Pro.

Prestazioni superiori a Xorg.

Ogni finestra effettua il rendering in maniera indipendente (**composizione**).

Sicurezza più elevata.

Lo stato interno di una finestra è invisibile alle altre (**isolamento**).

La composizione avviene nelle singole applicazioni, che eseguono con privilegi più bassi.

<https://www.secjuice.com/wayland-vs-xorg/>



Il logo di Wayland

I componenti di una GUI

(**Server**, window manager, toolkit, applicazione, graphical display manager)

Contro.

La stabilità di Wayland è ancora inferiore a quella di Xorg.

Wayland non funziona con alcuni modelli di schede video.

NVIDIA, I'm looking at you.

<https://wiki.gnome.org/Initiatives/Wayland/NVIDIA>

<https://www.youtube.com/watch?v=IVpOyKCNZYw>



Il logo di Wayland

I componenti di una GUI

(**Server**, window manager, toolkit, applicazione, graphical display manager)

Cosa scegliere?

Se supportato dalla propria scheda video, Wayland è preferibile a XOrg.

I componenti di una GUI

(Server, **window manager**, toolkit, applicazione, graphical display manager)

Window Manager.

Gestisce il disegno ed il posizionamento di elementi grafici, in primis **finestre**.

Finestra: area visuale dello schermo che
riceve eventi tramite dispositivi di input (mouse, tastiera);
associa gli eventi a funzioni da eseguire;
contiene un'area interna (in cui una applicazione disegna la propria GUI);
contiene una decorazione esterna che la delimita.

Invia richieste di disegno (o buffer di pixel) al server grafico.

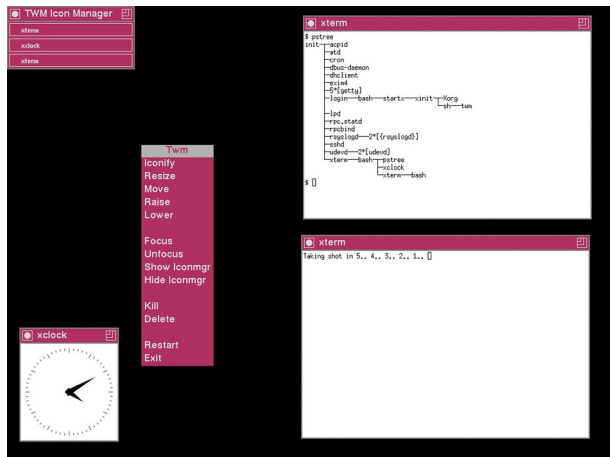
I componenti di una GUI

(Server, **window manager**, toolkit, applicazione, graphical display manager)

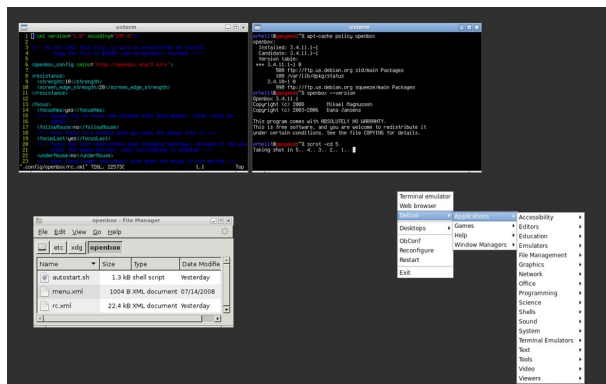
Window Manager “Stacking”.

Permette la sovrapposizione di finestre (**z-order**).

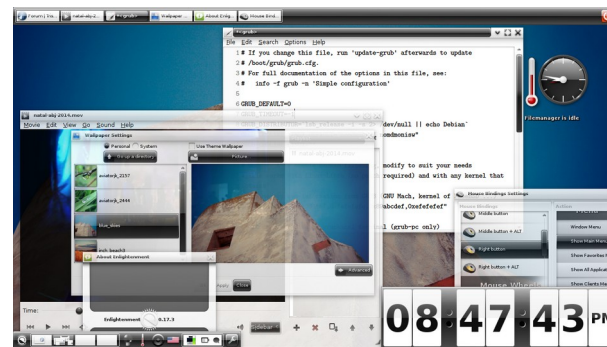
Le finestre sono disegnate sequenzialmente, una per una (anche se successivamente coperte da altre finestre).



Twm



Openbox



Enlightenment
E tanti altri...

I componenti di una GUI

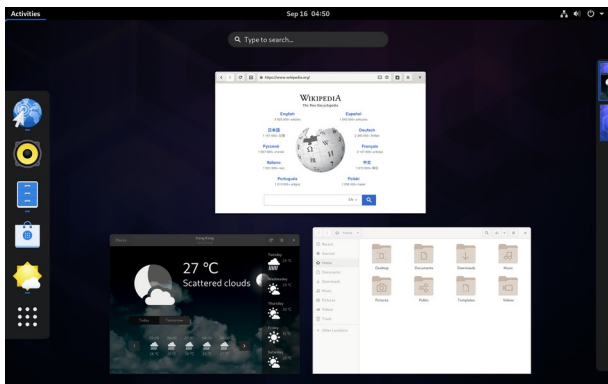
(Server, **window manager**, toolkit, applicazione, graphical display manager)

Window Manager “Compositing”.

Ogni finestra si ridisegna indipendentemente.

Uso intensivo di effetti 2D e 3D (accelerati in hardware).

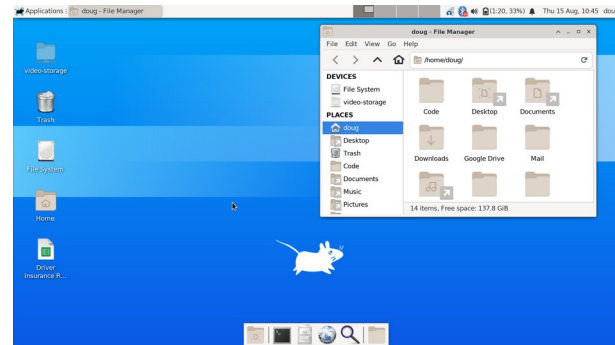
Prestazioni migliori rispetto ai window manager stacked.



Mutter



KWin



Xfwm

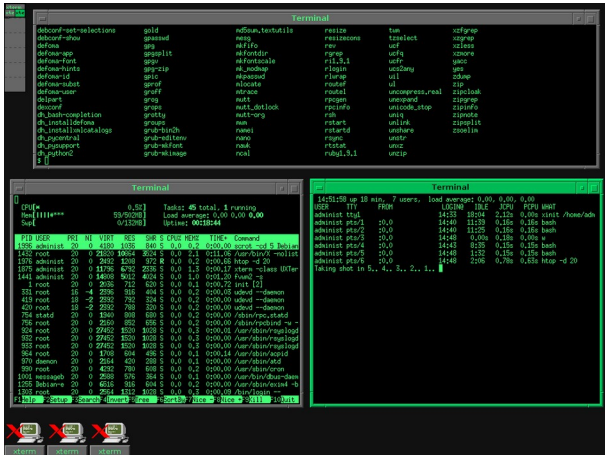
E tanti altri...

I componenti di una GUI

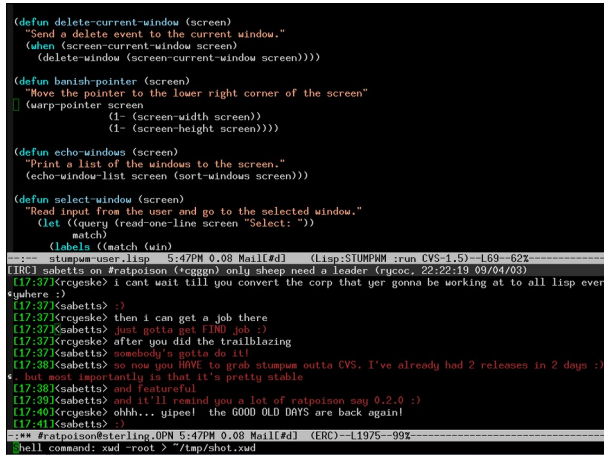
(Server, **window manager**, toolkit, applicazione, graphical display manager)

Window Manager “Dynamic”.

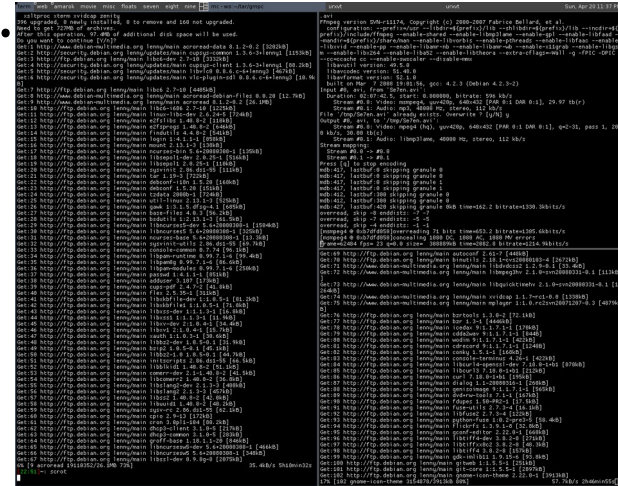
È un window manager tiling in cui la copertura dello schermo tramite finestre (orizzontale, verticale, finestra in evidenza) può essere scelta dall'utente.



FVWM



xmonad



Awesome
E tanti altri...

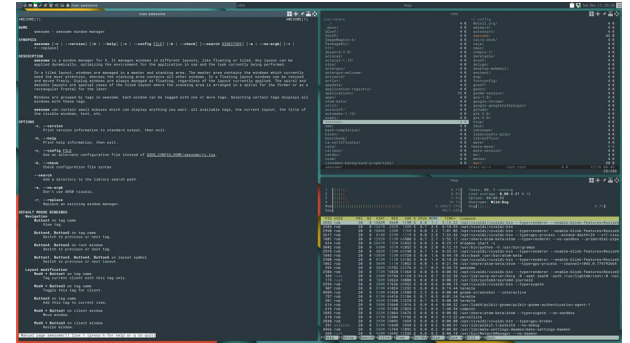
I componenti di una GUI

(Server, **window manager**, toolkit, applicazione, graphical display manager)

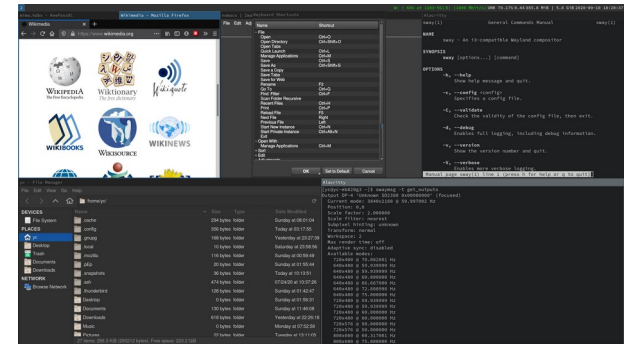
Alcuni window manager possono ricadere in categorie distinte!

i3 è un window manager tiling e dinamico!

Sway (fork di i3) è un window manager tiling, dinamico e per giunta compositing!



i3 (layout utente)



Sway (layout utente) 34

I componenti di una GUI

(Server, window manager, **toolkit**, applicazione, graphical display manager)

Cosa scegliere?

È questione di gusti (your mileage may vary).

Tendenze:

Utenti alle prime armi → Stacking.

Burocrati consolidati → Stacking.

Programmatori esperti → Tiling o dynamic.

Gamer incalliti → Compositing.

Esperti multimediali → Compositing.

I componenti di una GUI

(Server, window manager, **toolkit**, applicazione, graphical display manager)

Toolkit grafico.

È un insieme di applicazioni e librerie per la creazione di interfacce grafiche dal look omogeneo.

Le funzioni di libreria inviano richieste di disegno (o buffer di pixel) al server grafico.

Le funzioni di libreria possono essere usate da diversi linguaggi di programmazione tramite opportuni **binding**.

Paradigma di programmazione: ad oggetti, ad eventi.

I componenti di una GUI

(Server, window manager, **toolkit**, applicazione, graphical display manager)

Toolkit GTK.

Progetto iniziato nel 1998.

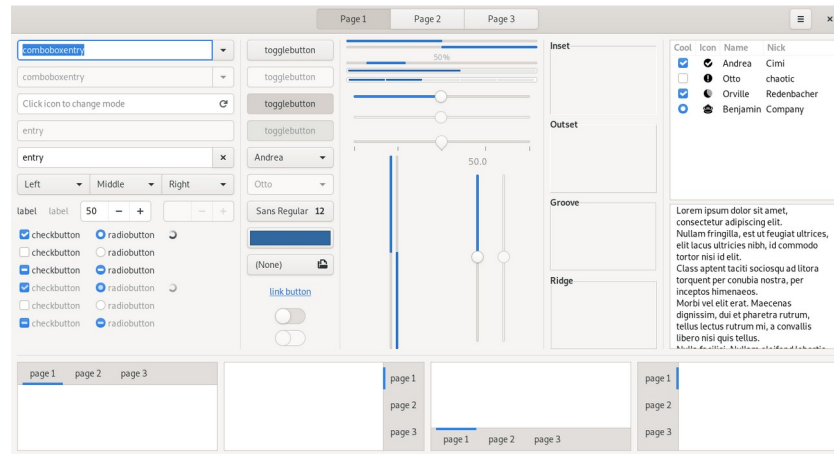
Scritto in C.

Concepito per Gnu Image Manipulation Program (GIMP).

Libreria di elementi grafici (**widget**).

Libreria di funzioni di utilità.

Applicazione di design GUI (**Glade**).



Widget GTK

I componenti di una GUI

(Server, window manager, **toolkit**, applicazione, graphical display manager)

Pro.

Prestazioni elevate.

Consumo di risorse ridotto.

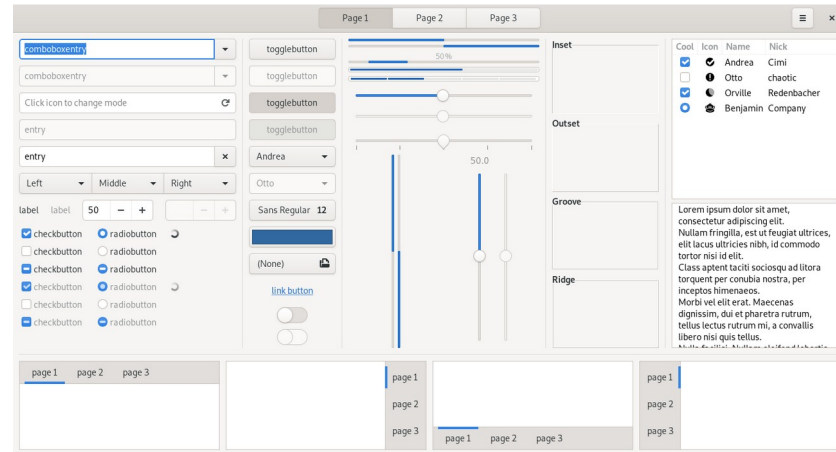
Estraneamente popolare.

Contro.

API scritta in un linguaggio non ad oggetti (più ostica).

Non portabile al di fuori di UNIX.

Gli sviluppatori non sono sempre recettivi alle richieste degli utenti.



Widget GTK

I componenti di una GUI

(Server, window manager, **toolkit**, applicazione, graphical display manager)

Toolkit Qt.

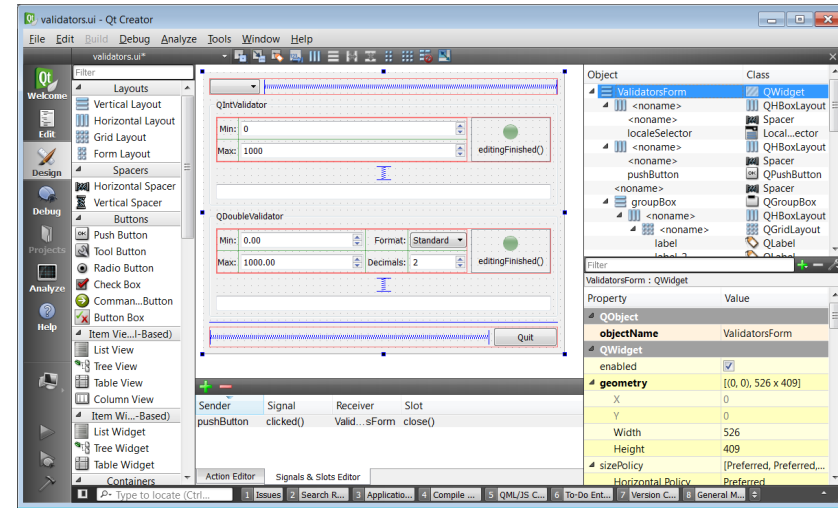
Progetto iniziato nel 1991.

Scritto in C++.

Concepito per fornire uno standard multi-piattaforma.

Libreria di elementi grafici (**widget**) e funzioni di utilità.

Applicazione di design GUI (**Qt Creator**).



Widget Qt

I componenti di una GUI

(Server, window manager, **toolkit**, applicazione, graphical display manager)

Pro.

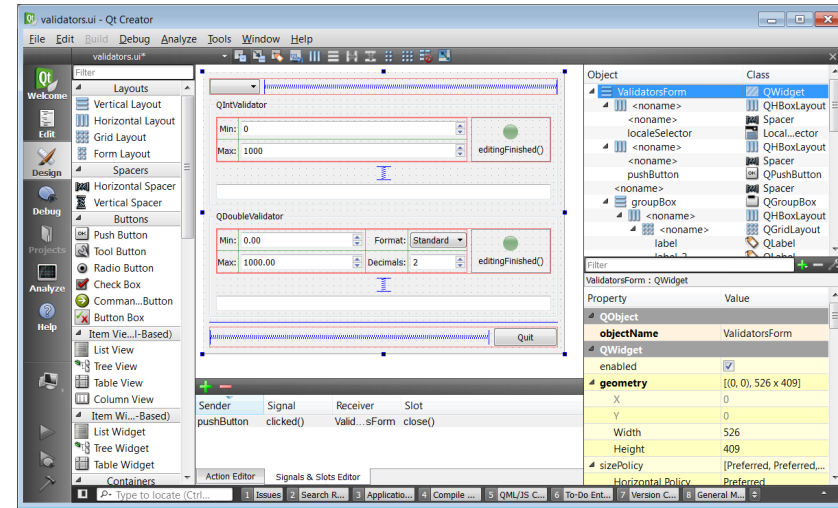
Portabilità.

API ad oggetti, più ricca rispetto a GTK.

Contro.

Prestazioni peggiori rispetto a GTK.

Consumo di risorse elevato.



Widget Qt

I componenti di una GUI

(Server, window manager, **toolkit**, applicazione, graphical display manager)

Toolkit Motif.

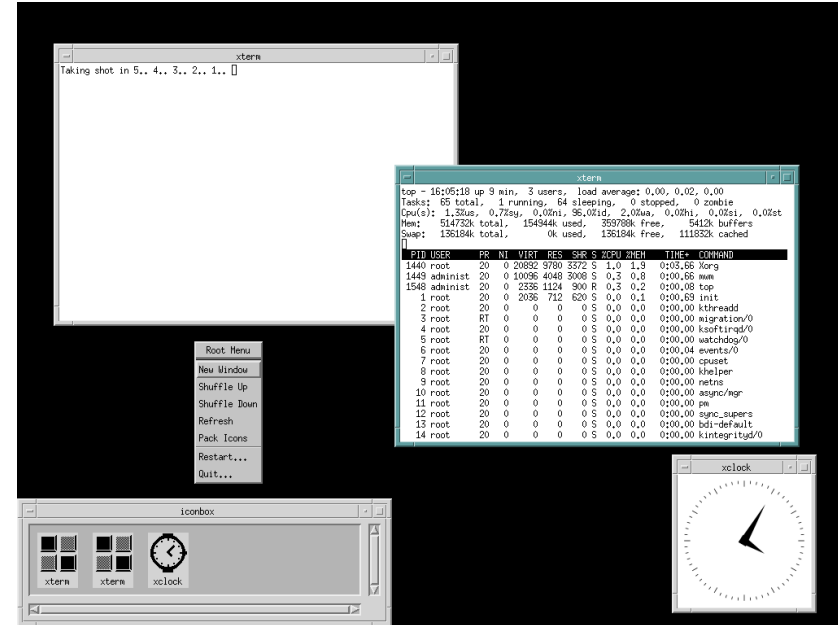
Progetto iniziato nel 1989.

Scritto in C.

Concepito per fornire uno standard per le workstation UNIX.

Clone di OPENLOOK (la prima GUI UNIX).

Libreria di elementi grafici (**widget**).



Widget Motif

I componenti di una GUI

(Server, window manager, **toolkit**, applicazione, graphical display manager)

Pro.

Prestazioni elevate.

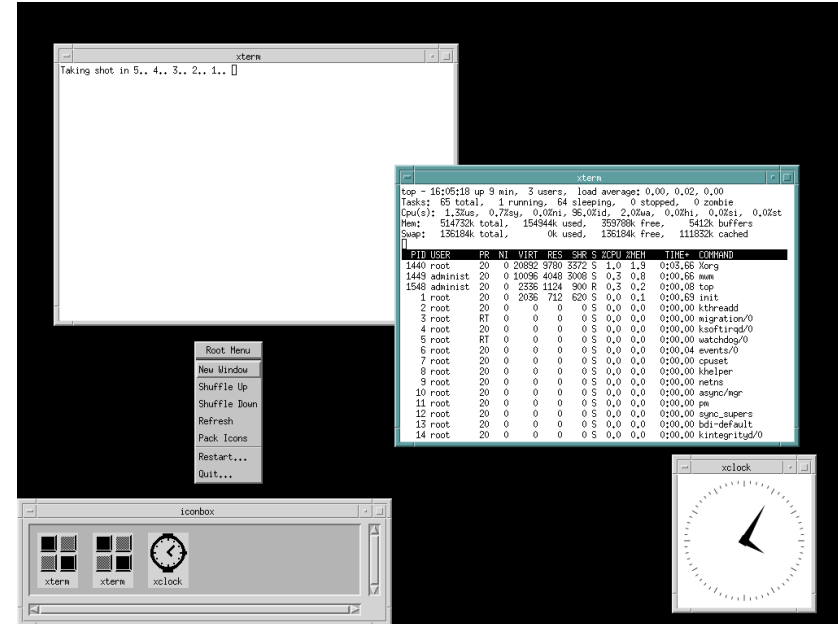
Consumo di risorse basso.

Contro.

Funzionalità ridotte rispetto a

GTK e Qt.

Obsoleto.



Widget Motif

I componenti di una GUI

(Server, window manager, **toolkit**, applicazione, graphical display manager)

Cosa scegliere?

Se le risorse hardware del calcolatore sono esigue, è preferibile GTK.

Se la portabilità tra diversi SO è un must, è preferibile Qt.

Per il resto, è questione di gusti (your mileage may vary).

I componenti di una GUI

(Server, window manager, toolkit, **applicazione**, graphical display manager)

Applicazione.

Tipicamente, è un wrapper grafico a funzionalità esistenti (applicazioni da linea di comando, librerie esterne). Usa un toolkit per creare una GUI dall'aspetto omogeneo.

Categorie di applicazioni disponibili:

- sistema (terminali, gestione periferiche, installazione software, etc.).

- editor di testo.

- internet (browser Web, client di posta elettronica, etc.).

- multimediale (riproduttori audio, video, immagini).

- svago (giochi).

- documentazione (manuali in linea).

- ufficio (documenti, fogli di calcolo, presentazioni).

I componenti di una GUI

(Server, window manager, toolkit, **applicazione**, graphical display manager)

Struttura visiva.

Dipende fortemente:

dal tipo di applicazione;

dal window manager adottato dall'utente;

dal toolkit grafico utilizzato nell'implementazione.

Sono **sempre** presenti:

una **finestra** (disegnata dal window manager);

un'**area interna** alla finestra (gestita dall'applicazione).

In generale, **possono** essere presenti:

menu per la scelta delle funzionalità;

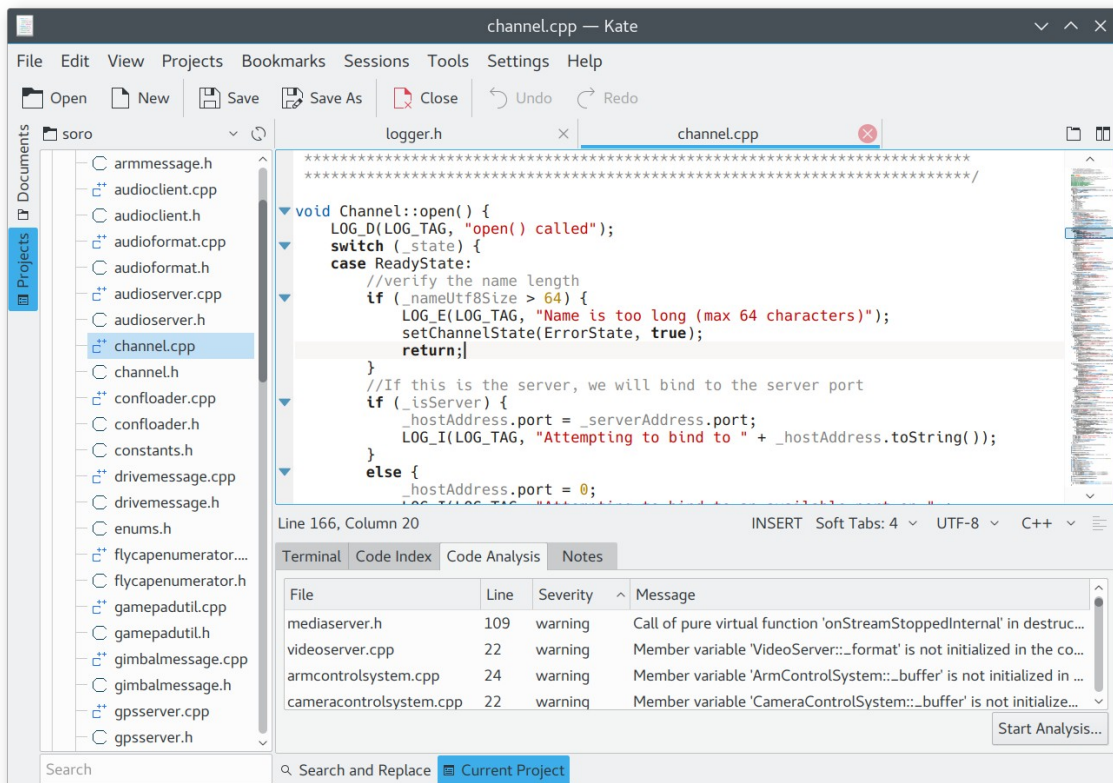
icone per la scelta rapida delle funzionalità più comuni;

riga di stato (per segnalare lo stato interno di un documento).

I componenti di una GUI

(Server, window manager, toolkit, **applicazione**, graphical display manager)

Editor di testo
"Kate".

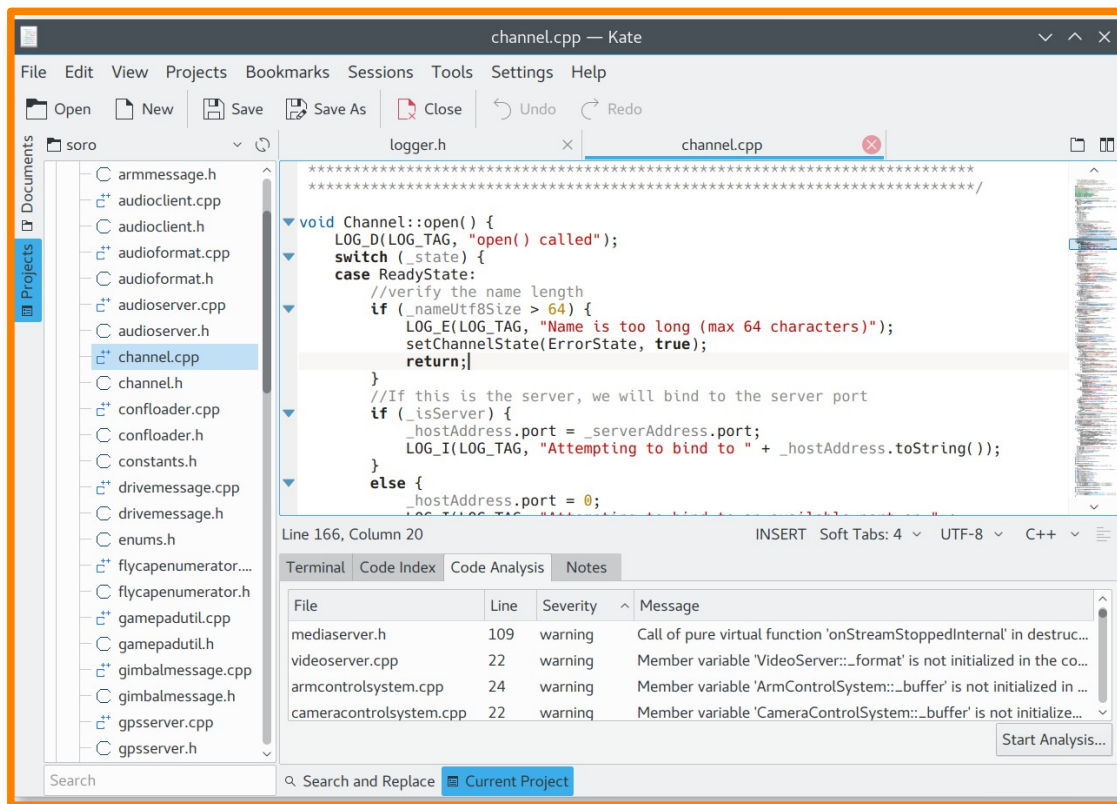


I componenti di una GUI

(Server, window manager, toolkit, **applicazione**, graphical display manager)

Editor di testo
"Kate".

Finestra.

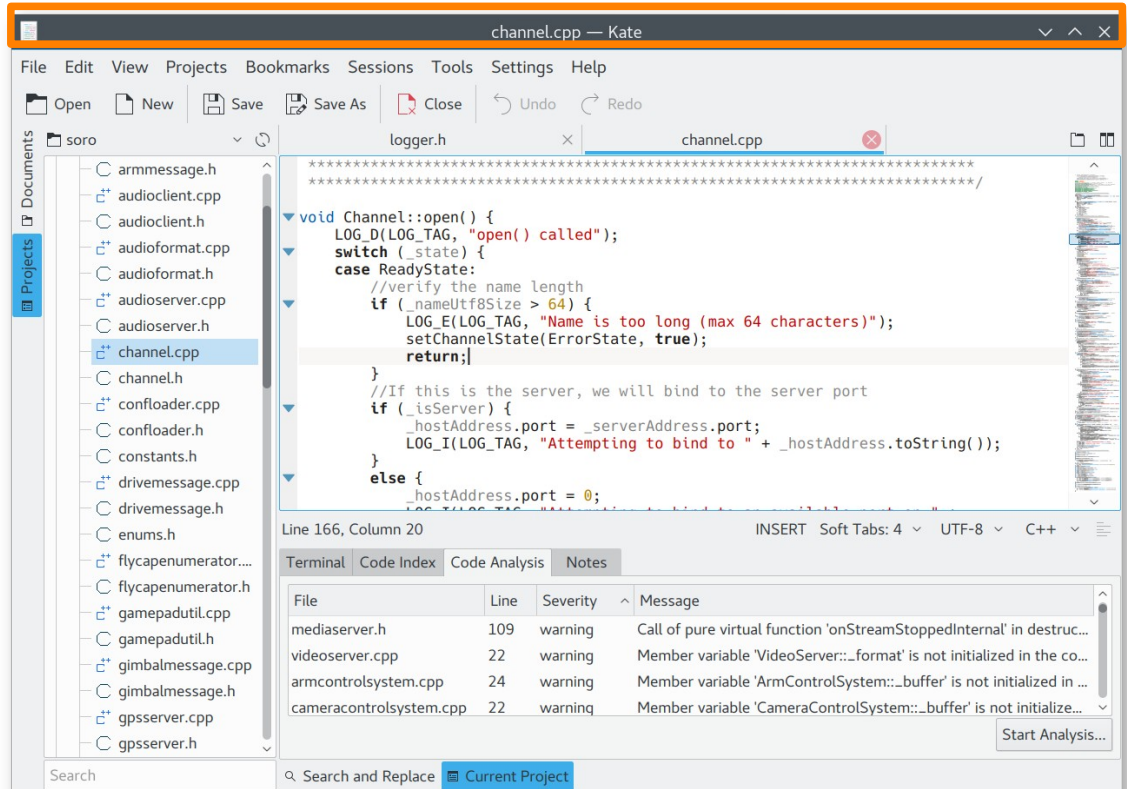


I componenti di una GUI

(Server, window manager, toolkit, **applicazione**, graphical display manager)

Editor di testo
“Kate”.

Intestazione
della finestra.
Titolo.
Operazioni
sulla finestra.

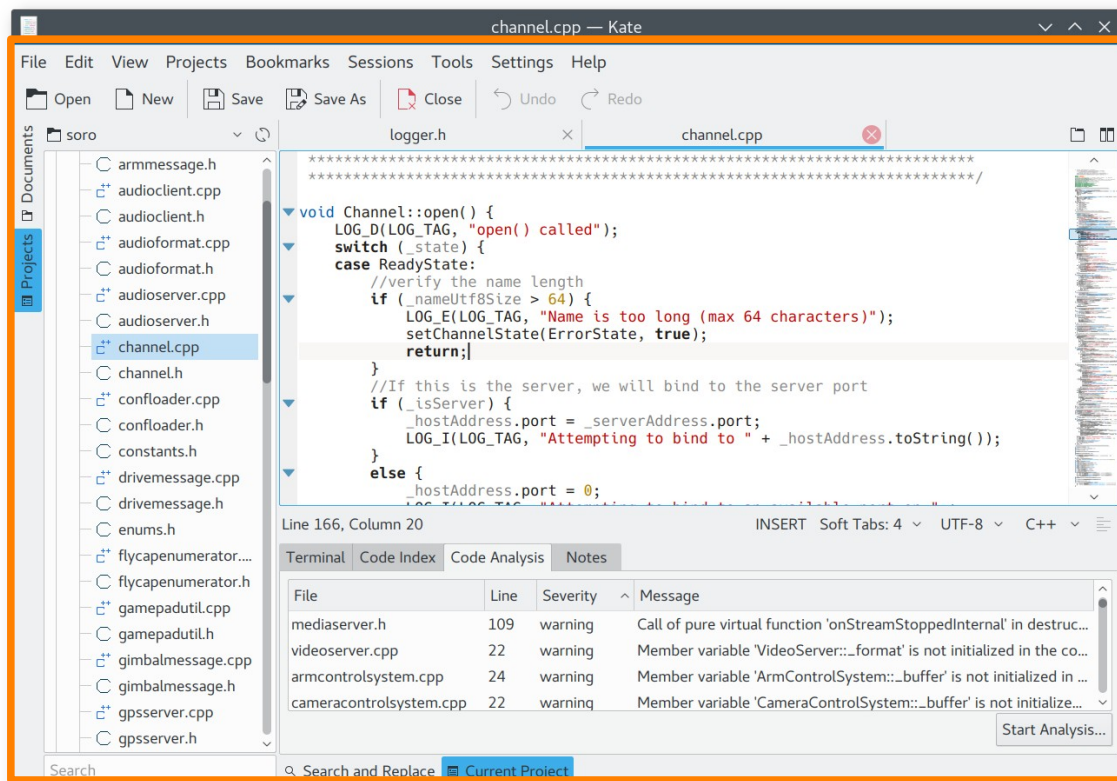


I componenti di una GUI

(Server, window manager, toolkit, **applicazione**, graphical display manager)

Editor di testo
"Kate".

Area interna alla
finestra.

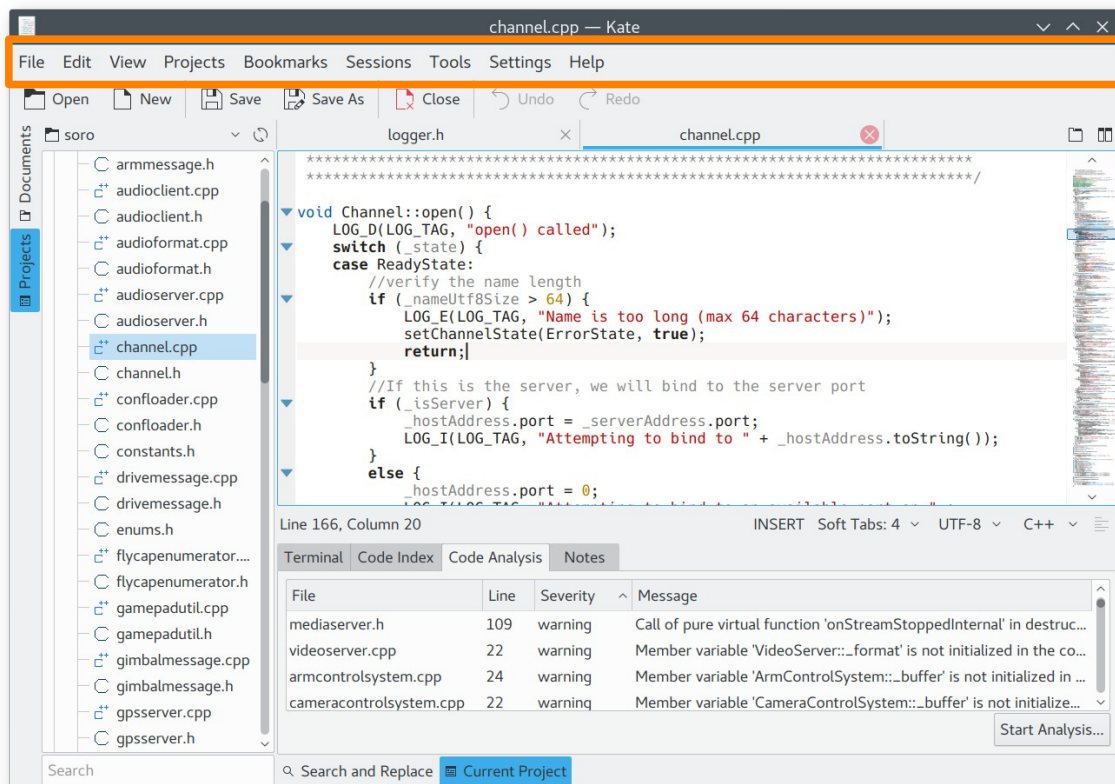


I componenti di una GUI

(Server, window manager, toolkit, **applicazione**, graphical display manager)

Editor di testo
"Kate".

Menu testuali.

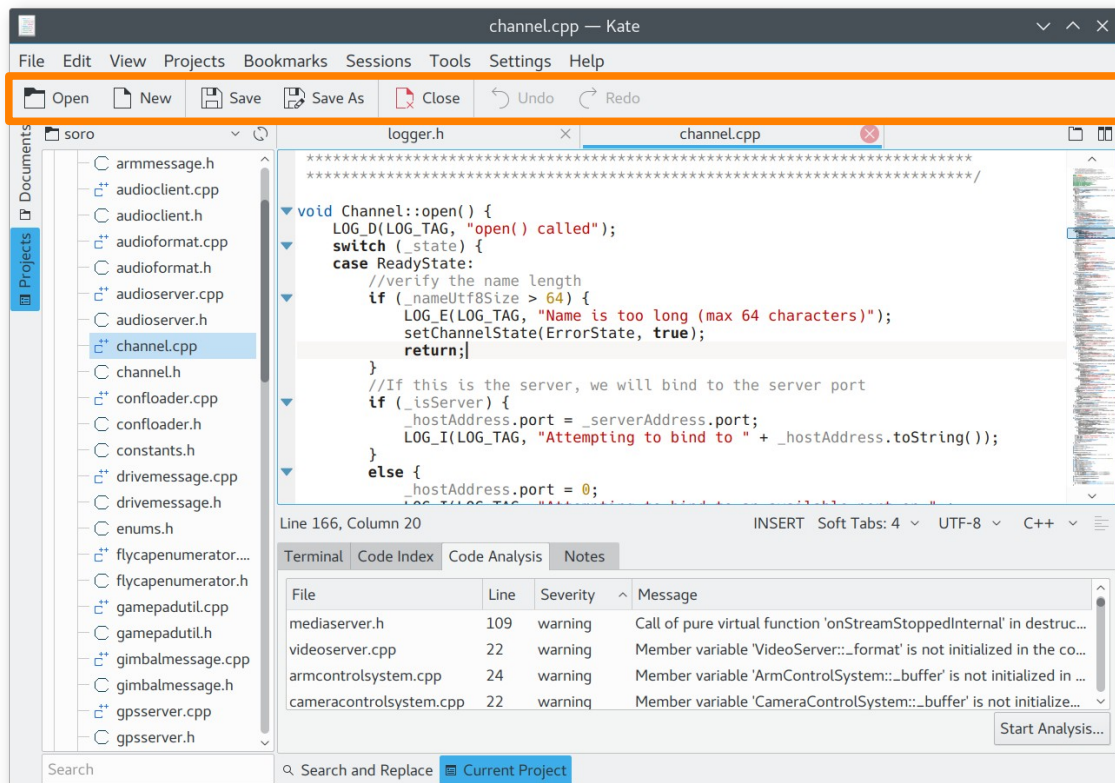


I componenti di una GUI

(Server, window manager, toolkit, **applicazione**, graphical display manager)

Editor di testo
"Kate".

Icone.

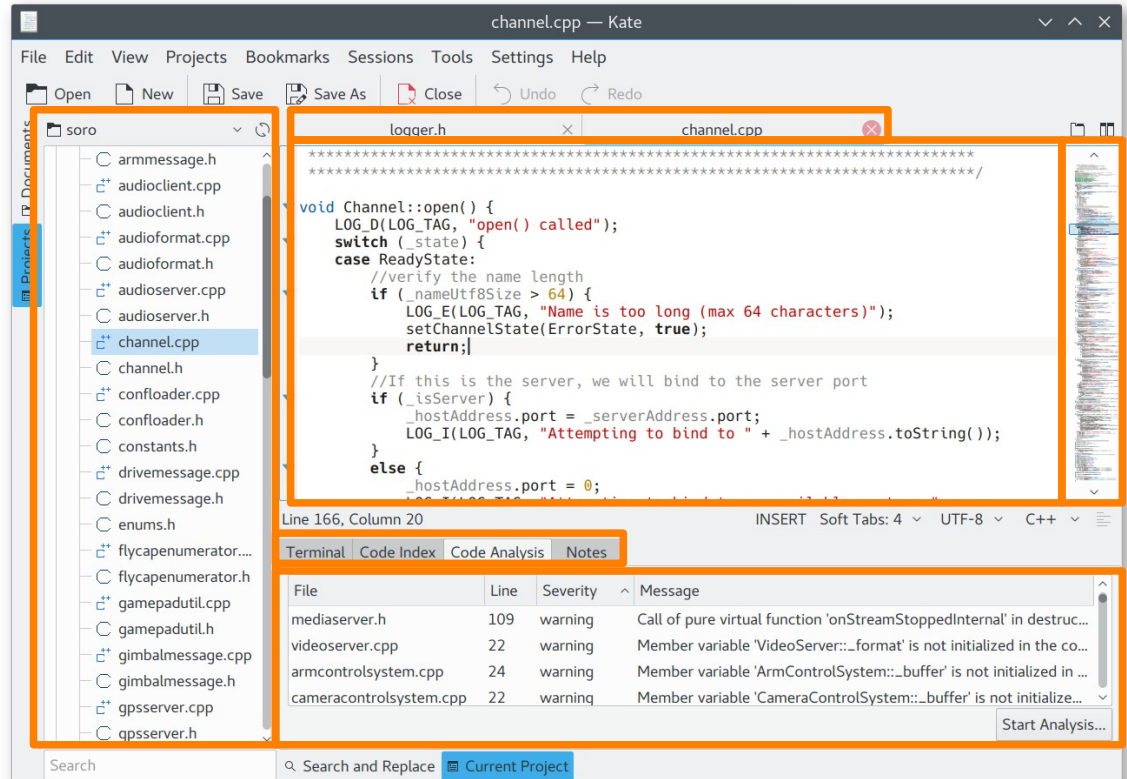


I componenti di una GUI

(Server, window manager, toolkit, **applicazione**, graphical display manager)

Editor di testo
"Kate".

Widget grafici
specifici alla lo-
gica applicativa.

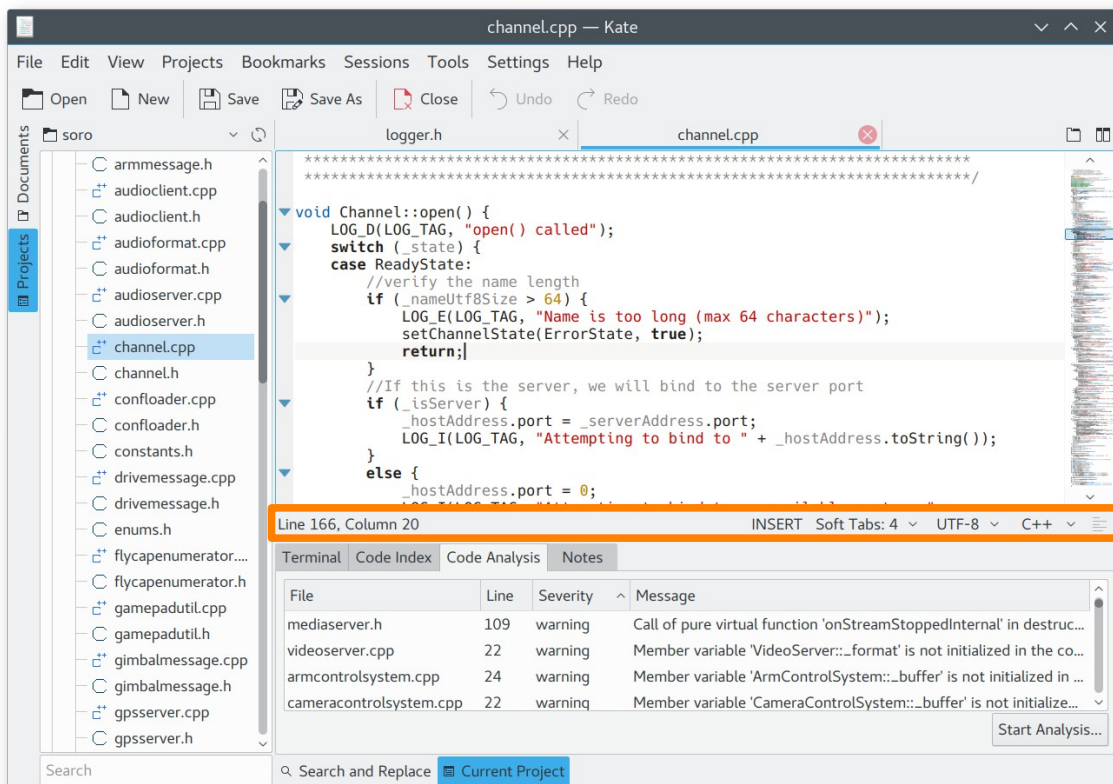


I componenti di una GUI

(Server, window manager, toolkit, **applicazione**, graphical display manager)

Editor di testo
"Kate".

Riga di stato.



I componenti di una GUI

(Server, window manager, toolkit, applicazione, **graphical display manager**)

Graphical Display Manager.

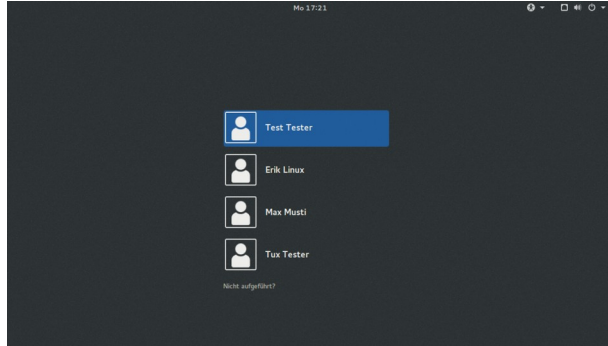
Fornisce un meccanismo di autenticazione (tipicamente, basato su username e password).

Esegue le applicazioni necessarie per inizializzare una sessione di lavoro (in primis, il window manager).

Permette la selezione di uno tra i molteplici window manager installati.

I componenti di una GUI

(Server, window manager, toolkit, applicazione, **graphical display manager**)



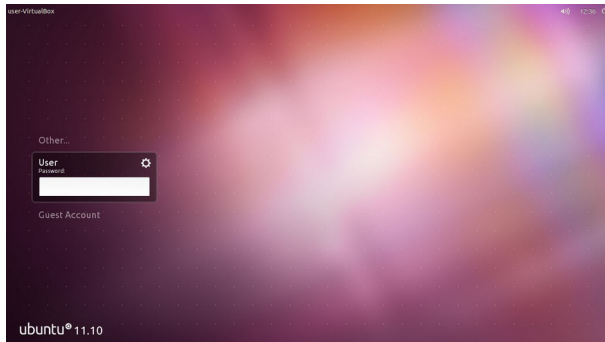
GDM



LXDM



KDM



LightDM

E tanti altri...

I componenti di una GUI

(Server, window manager, toolkit, applicazione, **graphical display manager**)

Cosa scegliere?

È questione di gusti (your mileage may vary).
Sono grossomodo tutti equivalenti.

Assemblaggio di un ambiente grafico

(Un compito estremamente arduo)

Per disporre di un ambiente grafico completo, in linea di principio un utente (anche alle prime armi) dovrebbe, da solo:

scegliere, scaricare, installare e configurare

server grafico;

window manager;

toolkit grafico;

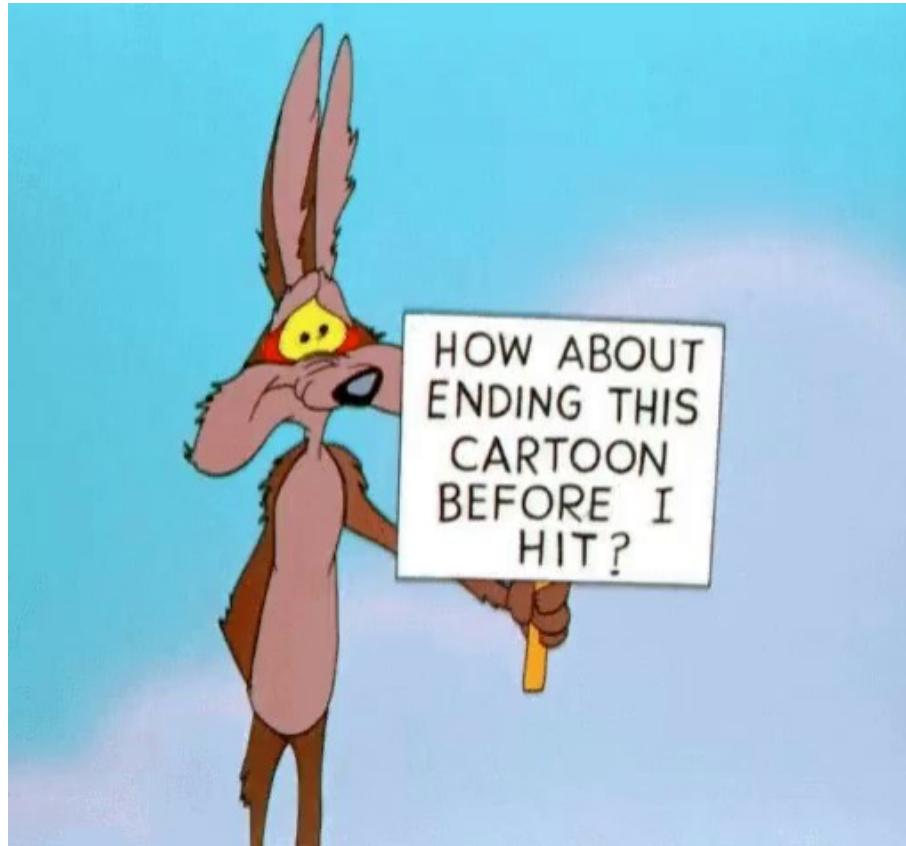
graphical display manager;

tutte le applicazioni grafiche necessarie ai propri scopi.

pregare \$DEITY che i vari componenti scelti siano compatibili tra loro nella realtà (e non solo sulla carta!).

Se vi sentite così...

(...avete la piena comprensione del docente!)



Ambiente desktop

(Una enorme semplificazione per l'utente)

Per risparmiare tale martirio agli utenti, gli sviluppatori delle interfacce grafiche mettono a disposizione veri e propri **ambienti desktop (desktop environment)**.

Ambiente desktop. È una collezione coesa, coerente, testata di graphical display manager, toolkit e applicazioni grafiche.

Obiettivi.

Fornire all'utente una "user experience" il più omogenea e uniforme possibile.

Semplificare l'installazione di un ambiente grafico completo.

Ambienti desktop principali

(GNOME, KDE, XFCE)

GNOME.

Progetto iniziato nel 1999.

Concepito per fornire una implementazione "free" di un ambiente desktop ad oggetti.

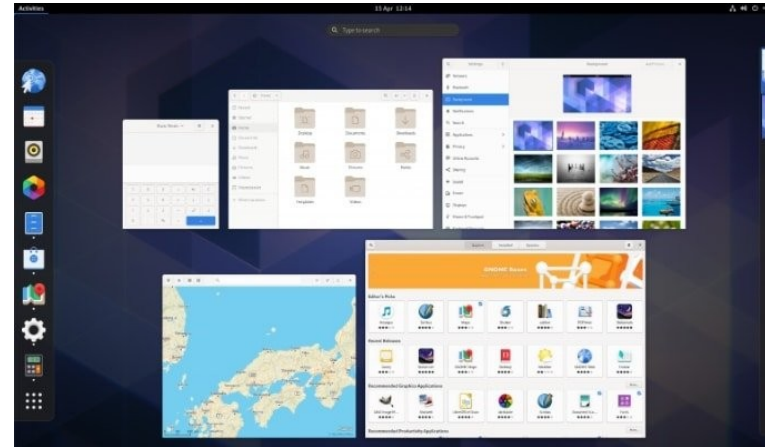
GNU Network Object Model Environment.

Graphical display manager: GDM.

Window manager: Mutter e GNOME shell.

Toolkit: GTK.

Server grafico: XOrg, Wayland.



GNOME 3.38

Ambienti desktop principali

(GNOME, KDE, XFCE)

Pro.

User experience più intuitiva del classico paradigma “desktop” (supportata da studi scientifici internazionali).

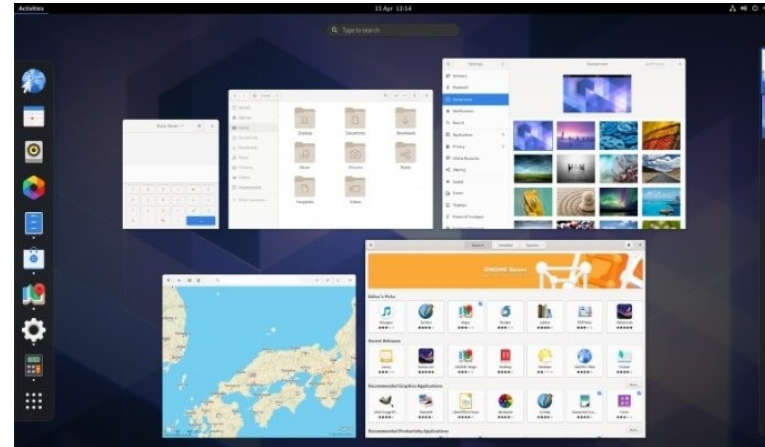
Stacked workspace.

Assenza di icone.

Pannello in alto.

Estendibile tramite plugin.

Consumo di risorse medio.



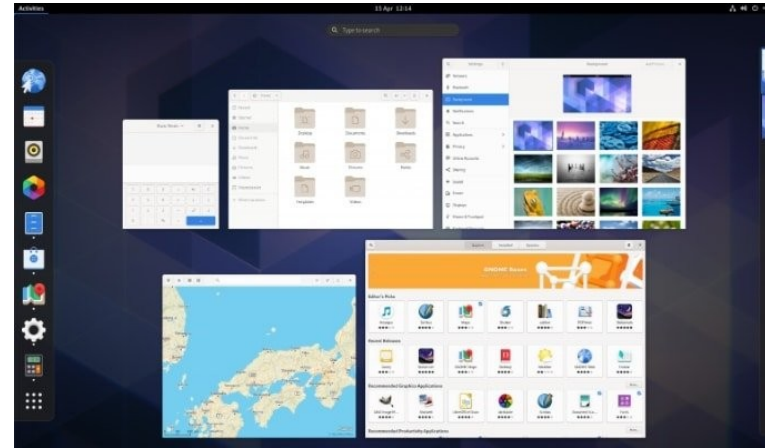
GNOME 3.38

Ambienti desktop principali

(GNOME, KDE, XFCE)

Contro.

Approccio minimalista (si tende a rimuovere la configurabilità e a fornire default validi per tutti). Gli sviluppatori non sono sempre recettivi alle richieste degli utenti.

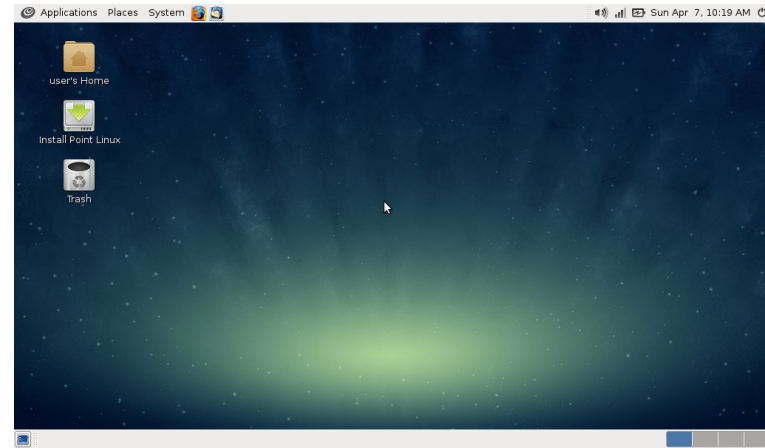


GNOME 3.38

Ambienti desktop principali

(GNOME, KDE, XFCE)

La nuova interfaccia di GNOME 3 non è stata accettata da tutti. L'interfaccia precedente (GNOME 2), un desktop classico, è stata usata per creare nuovi ambienti desktop meno "esoterici".



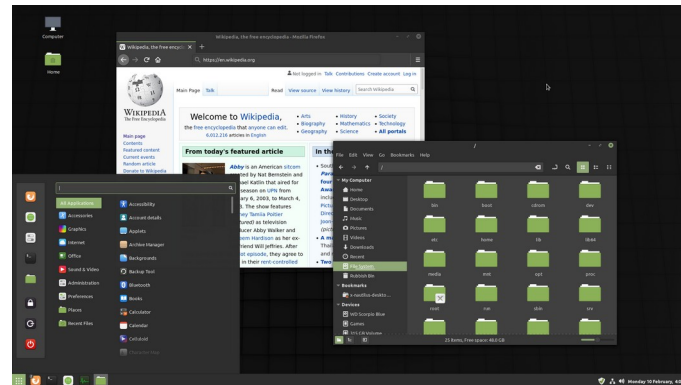
GNOME 2.32

Ambienti desktop principali

(GNOME, KDE, XFCE)

Cinnamon.

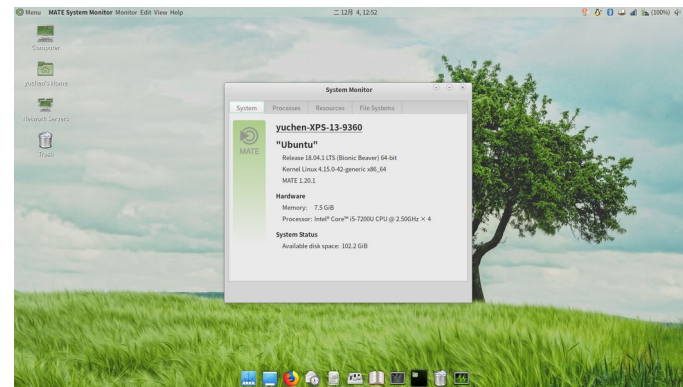
Window manager: Muffin.
Usa il codice di GNOME 3.
Ha l'aspetto di GNOME 2.
Usata in Linux Mint.



Cinnamon

Mate.

Window manager: marco.
Usa il codice di GNOME 2.
Ha l'aspetto di GNOME 2.
Usata in Linux Mint e Sabayon.



Mate

Ambienti desktop principali

(GNOME, **KDE**, XFCE)

KDE.

Progetto iniziato nel 1996.

Concepito per migliorare il desktop UNIX di riferimento, Common Desktop Environment.

Kool Desktop Environment.

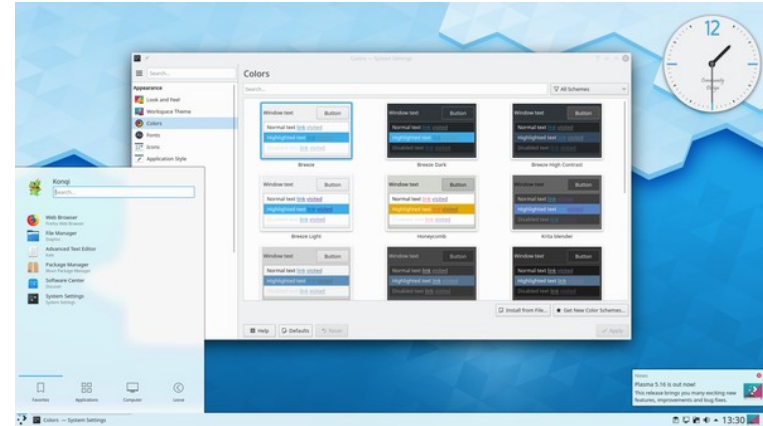
Graphical display manager: KDM.

Window manager: Kwin e

Plasma.

Toolkit: Qt.

Server grafico: XOrg, Wayland.



KDE Plasma 5.16

Ambienti desktop principali

(GNOME, **KDE**, XFCE)

Pro.

Funzionalità più ricca rispetto a GNOME.

Configurabilità migliore rispetto a GNOME.

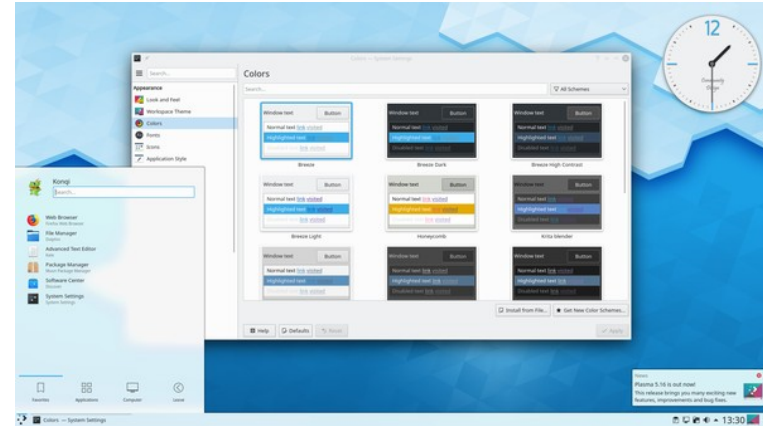
Effetti grafici esteticamente più appaganti rispetto a GNOME.

Estendibile tramite plugin.

Contro.

Consumo di risorse elevato.

Curva di apprendimento più elevata rispetto a GNOME.



KDE Plasma 5.16

Ambienti desktop principali

(GNOME, KDE, **XFCE**)

XFCE.

Progetto iniziato nel 1996.

Concepito per fornire un ambiente desktop UNIX-like leggero.

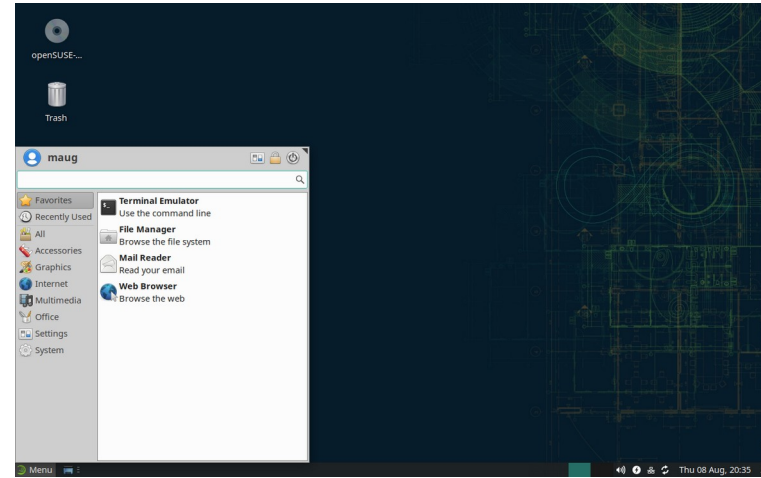
XForms **C**ommon **E**nvironment.

Graphical display manager: nessuno (LightDM è suggerito).

Window manager: Xfwm.

Toolkit: Xforms, poi GTK.

Server grafico: XOrg.



XFCE 4.14

Ambienti desktop principali

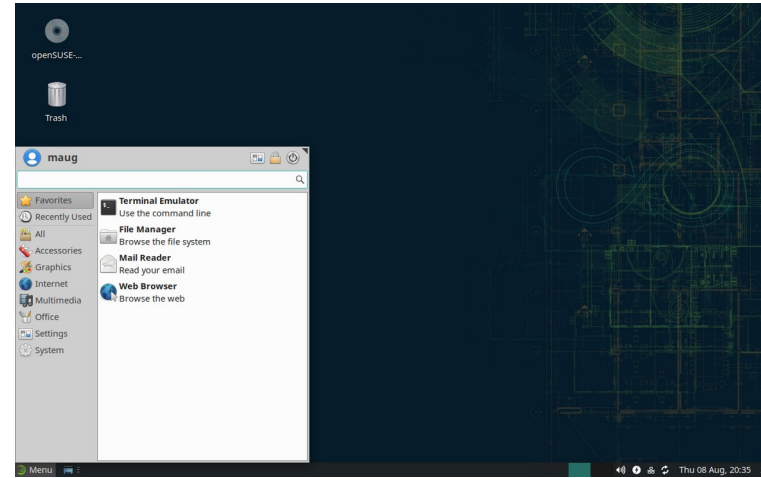
(GNOME, KDE, **XFCE**)

Pro.

Desktop classico, senza fronzoli.
Consumo di risorse basso.

Contro.

Non funziona su Wayland (Xfwm non è compositing).
Le funzionalità offerte sono decisamente inferiori rispetto a GNOME e a KDE.



XFCE 4.14

Supporto alle TUI

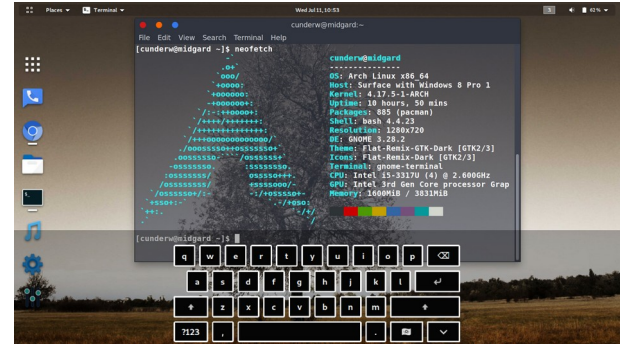
(Previsto in GNOME e KDE)

GNOME.

Supporta nativamente le gesture.
Testato su tablet (richiede un driver di input per touch screen).
Non ancora testato su smartphone.

KDE.

Supporta nativamente le gesture.
Testato su tablet (richiede un driver di input per touch screen).
Testato su smartphone (Plasma Mobile).



GNOME su OG Surface Pro



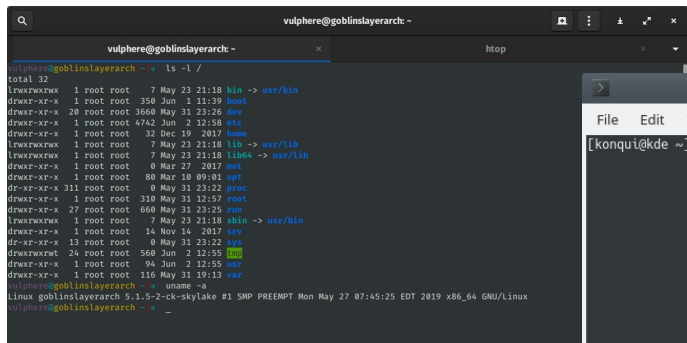
KDE su tablet Spark

CLI IN GNU/LINUX

Emulatore di terminale

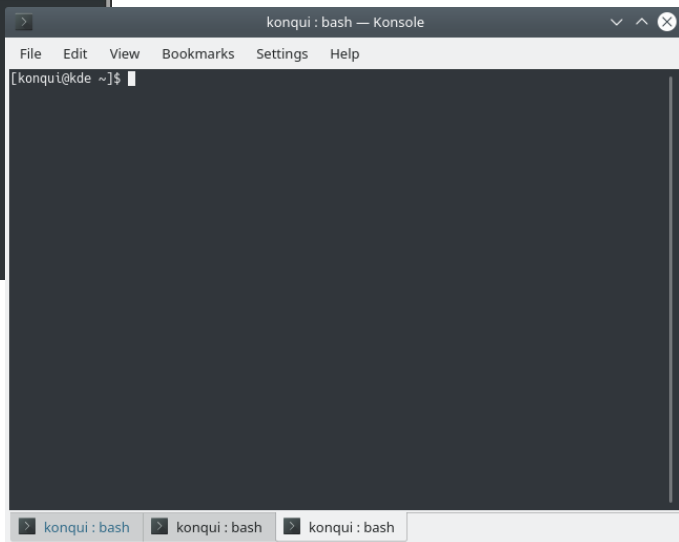
(Disponibile con un click o poco più)

Una delle applicazioni messe a disposizione dagli ambienti desktop è **l'emulatore di terminale**.



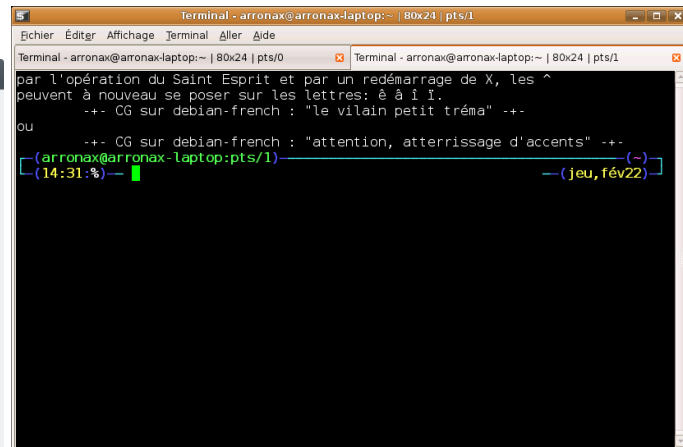
```
vulphere@goblnslayerarch:~$ ls -l /
total 32
lrwxrwxrwx 1 root root 7 May 23 21:18 bin -> /usr/bin
dwxr-xr-x 1 root root 350 Jun 1 11:59 boot
dwxr-xr-x 20 root root 3660 May 31 23:26 dev
dwxr-xr-x 1 root root 4742 Jun 2 12:58 etc
dwxr-xr-x 1 root root 32 Dec 19 2017 home
lrwxrwxrwx 1 root root 7 May 23 21:18 lib -> /usr/lib
lrwxrwxrwx 1 root root 7 May 23 21:18 lib64 -> /usr/lib
dwxr-xr-x 1 root root 0 Mar 27 2017 mnt
dwxr-xr-x 1 root root 80 Mar 10 09:01 opt
dr-xr-xr-x 311 root root 0 May 31 23:22 proc
dwxr-xr-x 1 root root 310 May 31 23:57 root
dwxr-xr-x 27 root root 600 May 31 23:25 run
lrwxrwxrwx 1 root root 7 May 23 21:18 sbin -> /usr/sbin
dwxr-xr-x 1 root root 14 Nov 16 2017 srv
dr-xr-xr-x 13 root root 0 May 31 23:22 sys
dwxrwxrwt 24 root root 560 Jun 2 12:55 tmp
dwxr-xr-x 1 root root 94 Jun 2 12:55 usr
dwxr-xr-x 1 root root 116 May 31 19:13 var
vulphere@goblnslayerarch:~$ uname -a
Linux goblnslayerarch 5.15-2-ck-skylake #1 SMP PREEMPT Mon May 27 07:45:25 EDT 2019 x86_64 GNU/Linux
vulphere@goblnslayerarch:~$
```

GNOME Terminal
(GNOME)



```
konqui: bash — Konsole
File Edit View Bookmarks Settings Help
[konqui@kde ~]$
```

Konsole (KDE)



```
Terminal - arronax@arronax-laptop:~ | 80x24 | pts/1
Fichier Éditer Affichage Terminal Aller Aide
Terminal - arronax@arronax-laptop:~ | 80x24 | pts/0
Terminal - arronax@arronax-laptop:~ | 80x24 | pts/1
par l'opération du Saint Esprit et par un redémarrage de X, les ^
peuvent à nouveau se poser sur les lettres: è à ï ÿ.
+- CG sur debian-french : "le vilain petit tréma" +-
ou
+- CG sur debian-french : "attention, atterrissage d'accents" +-
[aronax@arronax-laptop:pts/1]
(14:31:%)
--(jeu, fév22)--
```

XFCE Terminal
(XFCE)

Funzionalità offerte

(Riproduce in software il DEC VT-100 e simili)

L'emulatore di terminale riproduce le funzionalità di un terminale fisico (DEC VT-100 e simili).

Esecuzione di una applicazione interattiva (tipicamente, un interprete dei comandi).

Inoltro dell'input utente all'applicazione.

Stampa dell'output dell'applicazione.

Interpretazione di **sequenze di escape di terminale**.



DEC VT100
(1978)

Sequenze di escape di terminale? Eh?

(Whatcha talkin' about, Willis?)



Sequenza di escape di terminale

(Esecuzione di funzionalità terminale di basso livello)

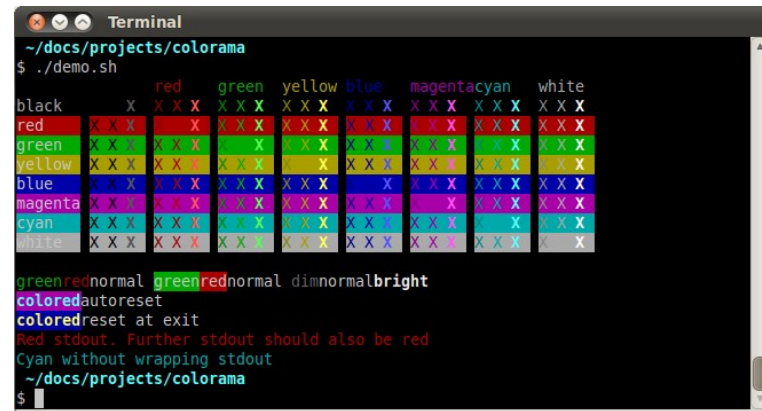
Una **sequenza di escape di terminale (terminal escape sequence)** è una sequenza di caratteri particolare che, se stampata, esegue una funzionalità di basso livello del terminale.

Cambio dell'aspetto dei caratteri (colori, effetto).

Cambio posizione del cursore.

Pulizia dello schermo.

La sequenza è spesso introdotta dal carattere ASCII 27 (Escape).



```
Terminal
~/docs/projects/colorama
$ ./demo.sh
black      x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x
red        x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x
green      x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x
yellow     x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x
blue       x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x
magenta    x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x
cyan       x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x
white      x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x  x

greenrednormal greenrednormal dimnormalbright
coloredautoreset
coloredreset at exit
Red stdout. Further stdout should also be red
Cyan without wrapping stdout
~/docs/projects/colorama
$
```

Output colorato tramite sequenze di escape

Un piccolo esperimento

(Cambio e ripristino del colore di primo piano)

Si apra un emulatore di terminale e si scriva il comando seguente:

```
echo -e "\033[32mtesto in verde\033[39m"
```

Si dovrebbe vedere la scritta "testo in verde" nel colore verde.

Un piccolo esperimento

(Cambio e ripristino del colore di primo piano)

Si apra un emulatore di terminale e si scriva il comando seguente:

```
echo -e "\033[32mtesto in verde\033[39m"
```



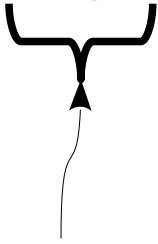
Il comando **echo**
stampa stringhe.

Un piccolo esperimento

(Cambio e ripristino del colore di primo piano)

Si apra un emulatore di terminale e si scriva il comando seguente:

```
echo -e "\033[32mtesto in verde\033[39m"
```



L'opzione `-e` abilita l'interpretazione delle sequenze di escape.

Un piccolo esperimento

(Cambio e ripristino del colore di primo piano)

Si apra un emulatore di terminale e si scriva il comando seguente:

```
echo -e "\033[32mtesto in verde\033[39m"
```



Queste sono due sequenze di escape.

Un piccolo esperimento

(Cambio e ripristino del colore di primo piano)

Si apra un emulatore di terminale e si scriva il comando seguente:

```
echo -e "\033[32mtesto in verde\033[39m"
```



Le sequenze iniziano con il carattere ASCII 27 (ESC), qui codificato in ottale (\033).

Un piccolo esperimento

(Cambio e ripristino del colore di primo piano)

Si apra un emulatore di terminale e si scriva il comando seguente:

```
echo -e "\033[32mtesto in verde\033[39m"
```


Il carattere [introduce le funzionalità di basso livello del terminale.

Un piccolo esperimento

(Cambio e ripristino del colore di primo piano)

Si apra un emulatore di terminale e si scriva il comando seguente:

```
echo -e "\033[32mtesto in verde\033[39m"
```



Il codice **32m** imposta il colore di primo piano a verde.

Un piccolo esperimento

(Cambio e ripristino del colore di primo piano)

Si apra un emulatore di terminale e si scriva il comando seguente:

```
echo -e "\033[32mtesto in verde\033[39m"
```



Questo testo è stampato con colore di primo piano verde.

Un piccolo esperimento

(Cambio e ripristino del colore di primo piano)

Si apra un emulatore di terminale e si scriva il comando seguente:

```
echo -e "\033[32mtesto in verde\033[39m"
```



Il codice **39m** imposta il colore di primo piano al valore di default.

Interprete da linea di comando

(Accetta comandi, li interpreta, li esegue, stampa l'output)

L'applicazione lanciata di default da un emulatore di terminale è l'**interprete da linea di comando** (**command line interpreter** o **shell**).

La shell implementa il **ciclo leggi-interpreta-stampa** (**read-evaluate-print loop**, o **REPL**):

- rimane in attesa di input utente fin quando non lo legge;
- interpreta l'input utente ed esegue le azioni corrispondenti;
- stampa l'output delle azioni eseguite.

Le shell principali

(sh, bash, dash, csh, tcsh, ksh, zsh, fish)

Bourne shell (sh).

Creata da Steven Bourne nel 1979.

Shell di default in UNIX V7.

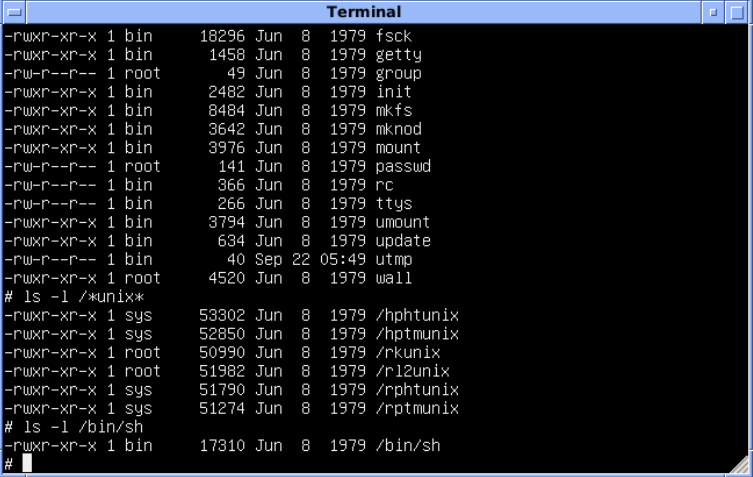
Madre (nonna?) di tutte le shell.

Segue rigorosamente lo standard POSIX (che definisce un SO UNIX).

Funzionalità ridotte.

Prestazioni elevate.

Oggi è implementata in bash (se eseguita con il nome "sh").



```
Terminal
-rwxr-xr-x 1 bin 18296 Jun 8 1979 fsck
-rwxr-xr-x 1 bin 1458 Jun 8 1979 getty
-rw-r--r-- 1 root 49 Jun 8 1979 group
-rwxr-xr-x 1 bin 2482 Jun 8 1979 init
-rwxr-xr-x 1 bin 8484 Jun 8 1979 mkfs
-rwxr-xr-x 1 bin 3642 Jun 8 1979 mknod
-rwxr-xr-x 1 bin 3976 Jun 8 1979 mount
-rw-r--r-- 1 root 141 Jun 8 1979 passwd
-rw-r--r-- 1 bin 366 Jun 8 1979 rc
-rw-r--r-- 1 bin 266 Jun 8 1979 ttys
-rwxr-xr-x 1 bin 3794 Jun 8 1979 umount
-rwxr-xr-x 1 bin 634 Jun 8 1979 update
-rw-r--r-- 1 bin 40 Sep 22 05:49 utmp
-rwxr-xr-x 1 root 4520 Jun 8 1979 wall
# ls -l /unix*
-rwxr-xr-x 1 sys 53302 Jun 8 1979 /hptunix
-rwxr-xr-x 1 sys 52850 Jun 8 1979 /hptmunix
-rwxr-xr-x 1 root 50990 Jun 8 1979 /rkunix
-rwxr-xr-x 1 root 51982 Jun 8 1979 /r12unix
-rwxr-xr-x 1 sys 51790 Jun 8 1979 /rptunix
-rwxr-xr-x 1 sys 51274 Jun 8 1979 /rptmunix
# ls -l /bin/sh
-rwxr-xr-x 1 bin 17310 Jun 8 1979 /bin/sh
#
```

Le shell principali

(sh, **bash**, dash, csh, tcsh, ksh, zsh, fish)

Bourne again shell (bash).

Creata da Brian Fox nel 1989.

Shell di default in GNU/Linux.

Ambiente di programmazione completo.

Prestazioni non elevatissime.

Implementa anche le funzionalità della Bourne shell.

```
vivek@nixcraft:/tmp$ help bg
bg: bg [job_spec ...]
    Move jobs to the background.

Place the jobs identified by each JOB_SPEC in the background, as if they
had been started with '&'. If JOB_SPEC is not present, the shell's notion
of the current job is used.

Exit Status:
Returns success unless job control is not enabled or an error occurs.
vivek@nixcraft:/tmp$ type -a cd
cd is a shell builtin
vivek@nixcraft:/tmp$
vivek@nixcraft:/tmp$ command -V ls
ls is aliased to `ls --color=auto'
vivek@nixcraft:/tmp$
```

Le shell principali

(sh, bash, **dash**, csh, tcsh, ksh, zsh, fish)

Debian Almquist shell (dash).

Creata da Herbert Xu nel 1997.

Port della shell ash, usata nel SO NetBSD.

Segue rigorosamente lo standard POSIX (che definisce un SO UNIX).

Basso consumo di risorse.

Prestazioni superiori a bash.

Non compatibile al 100% con bash.

Usata per eseguire script di avvio dei servizi.

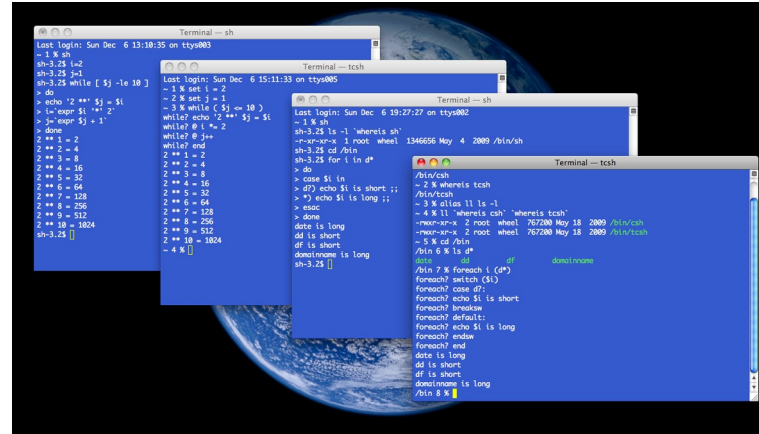
```
dave@howtogeek:~$ which sh
/bin/sh
dave@howtogeek:~$ ls -l /bin/sh
lrwxrwxrwx 1 root root 4 Sep 30 2019 /bin/sh -> dash
dave@howtogeek:~$
```

Le shell principali

(sh, bash, dash, **csh**, tcsh, ksh, zsh, fish)

C shell (csh).

Creata da Bill Joy nel 1979.
Shell di default in BSD UNIX.
Sintassi: C-like.
Basso consumo di risorse.
Prestazioni superiori a bash.
Incompatibile con bash.



Le shell principali

(sh, bash, dash, csh, **tcsh**, ksh, zsh, fish)

TENEX C shell (tcsh).

Creata da Ken Greer e altri nel 1983.

Shell di default del SO TENEX.

Estensione di csh.

Basso consumo di risorse.

Prestazioni superiori a bash.

Incompatibile con bash.

```
FreeBSD 11.0-STABLE (THINKPAD) #0 r317283: Sat Apr 22 23:16:56 EEST 2017
[ASCII art logo]
Welcome to FreeBSD!
Want to run the same command again?
In tcsh you can type "!!"
user % sduo pkg update
CORRECT>sudo pkg update (ylnle1a)?
```

Le shell principali

(sh, bash, dash, csh, tcsh, **ksh**, zsh, fish)

Korn shell (ksh).

Creata da David Korn nel 1983.

Shell di default del SO TENEX.

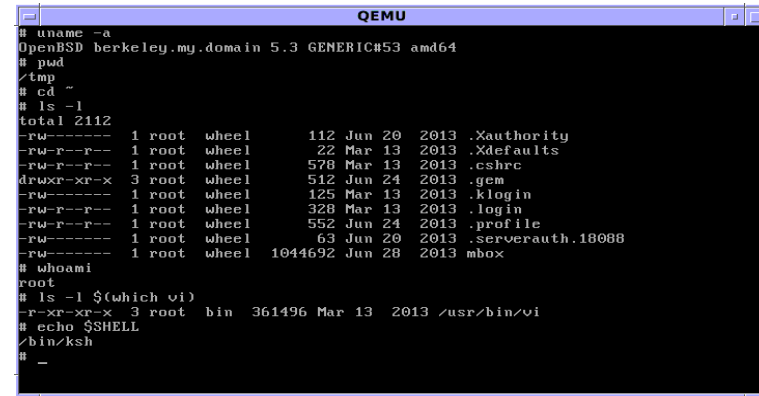
Basata su (e compatibile con) sh.

Segue rigorosamente lo standard POSIX (che definisce un SO UNIX).

Basso consumo di risorse.

Prestazioni superiori a bash.

Non compatibile al 100% con bash.



```
QEMU
# uname -a
OpenBSD berkeley.my.domain 5.3 GENERIC#53 amd64
# pwd
/tmp
# cd ~
# ls -l
total 2112
-rw----- 1 root  wheel   112 Jun 20  2013 .xauthority
-rw-r--r-- 1 root  wheel    22 Mar 13  2013 .xdefaults
-rw-r--r-- 1 root  wheel   578 Mar 13  2013 .cshrc
drwxr-xr-x 3 root  wheel   512 Jun 24  2013 .gem
-rw----- 1 root  wheel   125 Mar 13  2013 .klogin
-rw-r--r-- 1 root  wheel   328 Mar 13  2013 .login
-rw-r--r-- 1 root  wheel   552 Jun 24  2013 .profile
-rw----- 1 root  wheel    63 Jun 20  2013 .serverauth.10088
-rw----- 1 root  wheel 1044692 Jun 28  2013 mbox
# whoami
root
# ls -l $(which vi)
-r-xr-xr-x 3 root  bin  361496 Mar 13  2013 /usr/bin/vi
# echo $SHELL
/bin/ksh
#
```

Le shell principali

(sh, bash, dash, csh, tcsh, ksh, **zsh**, fish)

Z shell (zsh).

Creata da Paul Falstad nel 1990.

Estensione di bash.

Elevato consumo di risorse.

Prestazioni inferiori a bash.

Compatibile al 100% con bash.

Look grafico estendibile tramite temi.

```
~ > mkdir zshhh
~ > cd zshhh
~/zshhh > git init
Initialized empty Git repository in /home/michiel/zshhh/.git/
~/zshhh > master touch zshhh.txt
~/zshhh > master gaa
~/zshhh > master + gcam "first commit"
[master (root-commit) 40ef851] first commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 zshhh.txt
~/zshhh > master Zsh is great!
zsh: command not found: Zsh
✘ ~/zshhh > master
```

Le shell principali

(sh, bash, dash, csh, tcsh, ksh, zsh, **fish**)

Friendly interactive shell (fish).

Creata da Axel Liljencrantz nel 2005.

Funzionalità orientate al miglioramento dell'interattività e della "scopribilità" dei comandi.

Basso consumo di risorse.

Prestazioni superiori a bash.

Incompatibile con le altre shell.

```
Welcome to fish, the friendly interactive shell
Type help for instructions on how to use fish
shen@shelley -> ls
ls                               (List contents of directory)
lsattr (List file attributes on a Linux second extended file system)
lsb_release (Manual page for FSG lsb_release v1.4)
lshal      (List devices and their properties)
lshw       (List hardware)
lsmod      (Program to show the status of modules in the Linux Kernel)
lsmod.modutils (Rotten symbolic link, unknown)
lsopen     (List open files)
lspci      (List all PCI devices)
lspgpot    (Executable, 1.1kB)
lspnp     (List Plug and Play BIOS device nodes and resources)
lsusb     (List USB devices)
shen@shelley -> ls --s
--show-control-chars (Non graphic as-is) --size (Print size of files)
--si (Human readable size, base 1000) --sort= (Sort criteria)
shen@shelley -> ls --size | grep 4 # Search for small files
```


Comandi

(Opzione → Cosa fa il comando; Argomento → Su cosa opera il comando)

L'input ad una shell consiste di **comandi**.

La sintassi più generale di un comando UNIX è la seguente:

comando

comando <opzione>

comando <opzione> <argomento>

Opzione: specifica che cosa farà esattamente il comando.

Argomento: specifica su cosa opera l'opzione del comando.

Formato di opzioni e argomenti

(Formato breve e formato lungo)

Le opzioni sono specificate in due modi equivalenti.

Formato breve: un trattino seguito da un carattere alfanumerico (**-a**, **-h**, **-l**, **-x**, ...). Più opzioni possono essere unite (**-a -l** → **-al**).

Formato lungo: due trattini seguiti da una parola (**--all**, **--help**, **--list**, **--exclusive**, ...).

Gli argomenti sono stringhe generiche gestite dalla singola applicazione.

Opzioni e argomenti

(Un esempio concreto)

Esempio: si vogliono visualizzare tutti i file (nascosti e non) di una directory con relativi metadati. A tal scopo, si usa il comando `ls`. Nel seguito sono elencate alcune varianti equivalenti.

```
ls
```

```
ls
```

```
ls
```

```
-a -l
```

```
-al
```

```
--all -l
```

```
directory
```

```
directory
```

```
directory
```

Comando

Opzioni

Argomenti

La forma Backus-Naur

(La trovate nella documentazione in linea)

La sintassi del comando è descritta sinteticamente tramite il **formalismo di Backus-Naur**.

[stringhe]: le stringhe possono comparire 0 o 1 volta.

*{ stringhe }**: le stringhe possono comparire 0 o più volte.

{ stringhe }+: le stringhe possono comparire 1 o più volte.

str1/str2: può comparire str1 o str2.

-optlist: una o più opzioni brevi precedute da "-".

...: l'espressione precedente si ripete un numero arbitrario di volte.

La forma Backus-Naur del comando "ls"

(ls [OPTION]... [FILE]...)

La sintassi del comando `ls` è rappresentata nel modo seguente: `ls [OPTION] . . . [FILE] . . .`

È possibile specificare nessuna, una o più opzioni.

`ls, ls -a, ls -a -l.`

È possibile specificare nessuno, uno o più argomenti.

`ls, ls file1, ls file1 file2.`

È possibile combinare opzioni ed argomenti a piacere.

`ls, ls -a file1, ls -al file1 file2.`

Tipologie di comandi

(Interni, esterni)

Comando esterno (external command).

Fornito da un programma diverso dalla shell (BASH).
Memorizzato in una delle directory in PATH.
Quando è eseguito, la shell esegue una nuova applicazione.

Comando interno (shell builtin).

Comando fornito dalla shell.
Quando è eseguito, la shell esegue la funzione associata senza caricare altro.

Emulatore di terminale testuale

(Accessibile con CTRL-Alt-Fx)

L'emulatore di terminale in ambiente desktop non è l'unico modo di accedere ad una shell.

È disponibile anche un insieme di **emulatori di terminale testuali**, accessibili con la combinazione di tasti **<CTRL>-<Alt>-<Fx>**

(dove x=1, 2, ..., 6).

Uno degli emulatori è occupato dal server grafico.

```
Ubuntu 18.04.1 LTS ubuntu tty1
ubuntu login: root
Password:
Welcome to Ubuntu 18.04.1 LTS (GNU/Linux 4.15.0-36-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:   https://landscape.canonical.com
 * Support:      https://ubuntu.com/advantage

System information as of Wed Oct 24 12:14:37 AEDT 2018

System load:  0.08      Processes:      80
Usage of /:   32.0% of 9.7GB  Users logged in:  0
Memory usage: 11%      IP address for enp0s3: 192.168.1.131
Swap usage:   0%

 * Meltdown, Spectre and Ubuntu: What are the attack vectors,
  how the fixes work, and everything else you need to know
  - https://ubuntu.com/knownissues

13 packages can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

root@ubuntu:~#
```

Problema

(Non da poco)

Poiché l'emulatore di terminale testuale aggira completamente l'ambiente desktop, aggira pure l'autenticazione del graphical display manager!

Per evitare di usare un interprete dei comandi senza autenticazione preliminare, è necessario far precedere l'esecuzione dell'interprete dei comandi da una procedura di autenticazione.

Le applicazioni in gioco

(Emulatore di terminale testuale, gestore login, interprete comandi)

Emulatore di terminale testuale.

Implementazione equivalente all'emulatore di terminale grafico.

Nome storico: **getty**.

```
Ubuntu 18.04.1 LTS ubuntu tty1
ubuntu login: root
Password:
Welcome to Ubuntu 18.04.1 LTS (GNU/Linux 4.15.0-36-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

System information as of Wed Oct 24 12:14:37 AEDT 2018

System load:  0.08                Processes:            80
Usage of /:   32.0% of 9.78GB      Users logged in:    0
Memory usage: 11%                 IP address for enp0s3: 192.168.1.131
Swap usage:   0%

 * Meltdown, Spectre and Ubuntu: What are the attack vectors,
   how the fixes work, and everything else you need to know
   - https://ubu.one/u2Kknow

13 packages can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

root@ubuntu:~#
```

Le applicazioni in gioco

(Emulatore di terminale testuale, **gestore login**, interprete comandi)

Gestore di login.
Implementazione equivalente del graphical display manager.
Nome storico: **login**.

```
Ubuntu 18.04.1 LTS ubuntu tty1
ubuntu login: root
Password:
Welcome to Ubuntu 18.04.1 LTS (GNU/Linux 4.15.0-36-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Wed Oct 24 12:14:37 AEDT 2018

System load:  0.08          Processes:            80
Usage of /:   32.0% of 9.78GB Users logged in:       0
Memory usage: 11%          IP address for enp0s3: 192.168.1.131
Swap usage:   0%

 * Meltdown, Spectre and Ubuntu: What are the attack vectors,
   how the fixes work, and everything else you need to know
   - https://ubu.one/u2know

13 packages can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

root@ubuntu:~#
```

Le applicazioni in gioco

(Emulatore di terminale testuale, gestore login, **interprete comandi**)

Interprete dei comandi.
Interprete di default:
bash.

```
Ubuntu 18.04.1 LTS ubuntu tty1
ubuntu login: root
Password:
Welcome to Ubuntu 18.04.1 LTS (GNU/Linux 4.15.0-36-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Wed Oct 24 12:14:37 AEDT 2018

System load:  0.08          Processes:            80
Usage of /:   32.0% of 9.78GB Users logged in:       0
Memory usage: 11%          IP address for enp0s3: 192.168.1.131
Swap usage:   0%

 * Meltdown, Spectre and Ubuntu: What are the attack vectors,
   how the fixes work, and everything else you need to know
   - https://ubuntu.com/knownissues

13 packages can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

root@ubuntu:~#
```